Entwicklung einer E-Learning Anwendung für den Aufbau von Neutronenstreu-Experimenten am Hahn-Meitner-Institut Berlin GmbH

Diplomarbeit

im Studiengang
Medieninformatik
Schwerpunkt Medien
des Fachbereichs Informatik
der Technischen Fachhochschule Berlin
Vorgelegt von Andreas Tomm (s707580) im WS 2004
Betreuende Lehrkraft: Prof. des. Dr. Robert Strezebkowski

Inhaltsverzeichnis

1 Einführung 4
1.1 Einleitung 4
1.2 Motivation und Zielsetzung 5
1.3 Das Hahn-Meitner-Institut Berlin GmbH als Auftraggeber 5
1.4 Aufgabenstellung 6
2 Pflichtenheft 7
2.1 Zielbestimmung 7
2.2 Produkteinsatz 8
2.3 Produktumgebung 9
2.4 Produktübersicht 9
2.5 Produktfunktionen 10
2.6 Produktdaten 15
2.7 Produktleistungen 15
2.8 Benutzeroberfläche 16
2.9 Testfälle 16
2.10 Entwicklungsumgebung 16
3 Technisches Umfeld 17
3.1 Warum ein Autorensystem 17
3.2 Flash 19
3.3 Client-Server-Modell 21
3.4 XML 22
3.5 Flash und XML 25
3.6 T_EX 26
3.7 Wahl des Texteditors 28
4 Idee und Entwurf 29
4.1 Datenstrukturen und Inhalte 29
4.2 Die Oberfläche 31
5 Konzepte und Realisierung 35
5.1 Aufbau der XML-Dateien 36
5.2 Die Strukturierung in Flash 46
5.3 Die Programmierung in ActionScript 48
6 Installation und Anleitung 63

6.1 Benutzerhandbuch für den Anwender 63
6.2 Benutzerhandbuch für den Administrator 67
7 Fazit und Ausblick 73
8 Literatur und Quellenangaben 74
9 Anhang 75
9.1 Voreingestellte Sonderzeichen der Datei "sonderzeichen.xml" 75
9.2 Versuch ein XML-Objekt in einer Klasse zu generieren 76
9.3 Aufbau der Anwendung (Die Dateien) 77
9.4 CD-ROM - Diplom 87
9.5 CD-ROM – Fertige Anwendung 88
9.6 Danksagung 89

1 Einführung

1.1 Einleitung

Der Autor dieses Textes arbeitet als studentische Hilfskraft am Hahn-Meitner-Institut Berlin GmbH, im Folgenden HMI genannt. Das HMI ist eine Forschungseinrichtung mit einem 10 MW Reaktor, an welchem Neutronenstreu-Experimente durchgeführt werden. Das Gebäude, in dem der Reaktor und drum herum die Experimente angeordnet sind, heißt Experimentierhalle (im Folgenden E-Halle genannt), ein Gebäude weiter ist noch die Neutronen-Leiter Halle mit angeschlossen (Abbildung 1.1.1).

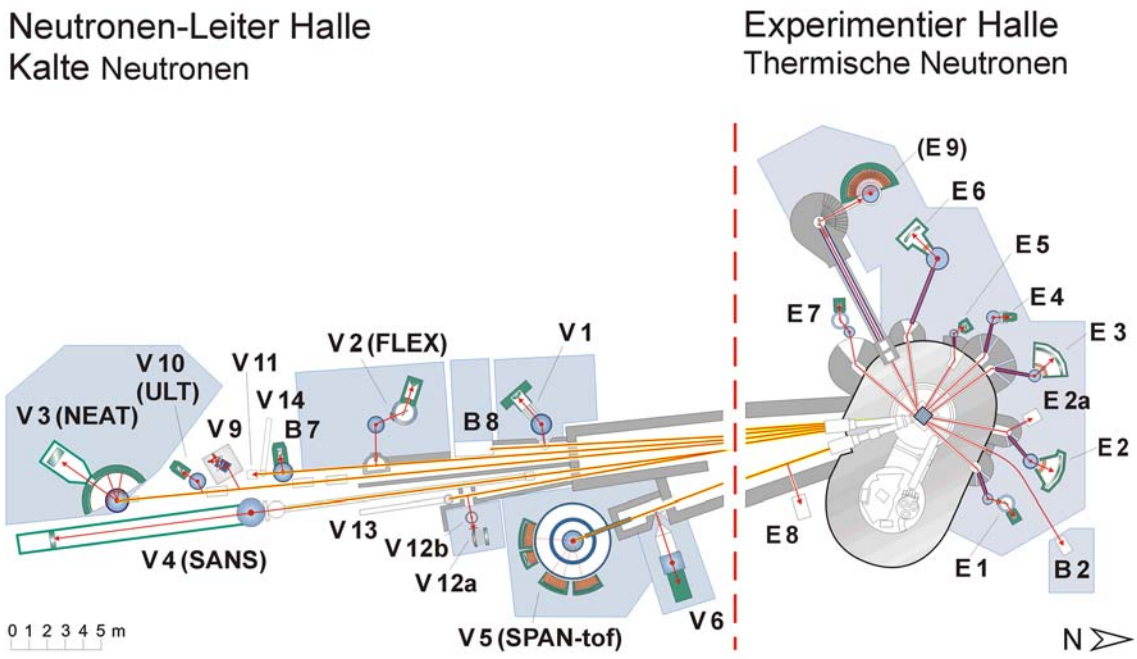


Abbildung 1.1.1: Rechts die E-Halle[1]

Dieses Projekt bezieht sich allerdings nur auf die E-Halle, kann aber auch für die Neutronen-Leiter Halle (V-Halle) genutzt werden.

Gäste aus aller Welt kommen nach Berlin zum HMI, um dort an einer mitgebrachten Probe zu messen. Diese Probe ist meistens eine Metalllegierung im Stück oder auch als Pulver. Sie wird an einem dieser Experimente mit Neutronen beschossen. Diese werden innerhalb der Probe abgelenkt, d.h. gestreut und werden dann wieder auf einer vorgegebenen Fläche, dem Detektor, aufgefangen und als Messwert gespeichert (siehe Abbildung 1.1.2).

Die Experimentaufbauten in der E- und V-Halle sind einzigartig und somit muss ein Experimentator, also der Experimentdurchführende, an jedem Experiment eingeführt werden, welches der jeweilige Experimentbetreuer übernimmt. Diese Einführung kann

[1] Quelle: Hahn-Meitner-Institut Berlin GmbH – http://www.hmi.de (Stand: 07.12.2004)

jedoch einige Zeit in Anspruch nehmen, da die Experimentaufbauten sehr kompliziert sind.

Was liegt also näher dem jeweiligen Gast, vorab eine Einführung mittels eines E-Learning-Tools zu geben. Dieses Tool sollte sowohl Offline auf einer CD-ROM und später auch Online über das Internet zur Verfügung stehen.

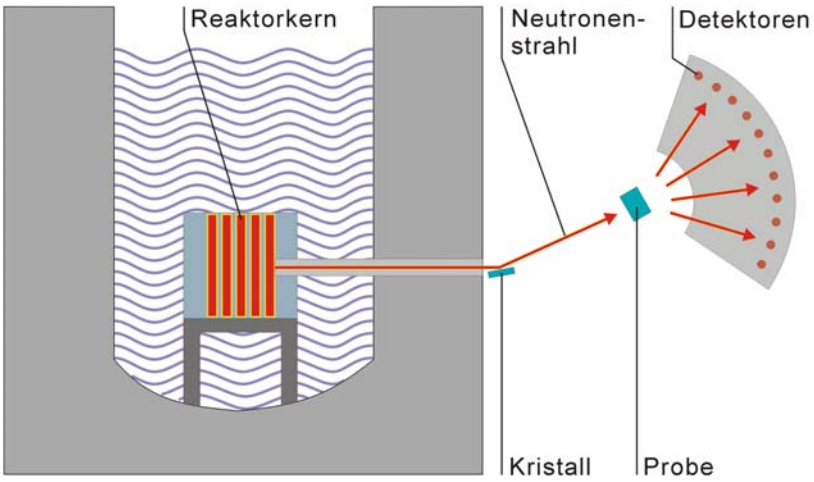


Abbildung 1.1.2: Streuprinzip[2]

1.2 Motivation und Zielsetzung

Da die Einführung an den Experimenten eine gewisse Zeit in Anspruch nimmt, soll für den Experimentverantwortlichen diese Zeit durch ein E-Learning-Tool verkürzt werden. Der Experimentator wird also vorab über den jeweiligen Experimentaufbau aufgeklärt und kann sich ausführlich informieren.

Grund für die Entscheidung ein solches Tool zu erstellen, sind die enormen Kosten für die Messzeit an einem Experiment. Ein Experiment dauert im Normalfall einige Stunden oder Tage, wenn der Experimentator davor noch eine ausgiebige Einführung am Experiment bekommt, so verlängert sich die Gesamtzeit, d.h. Einführung des Experimentatoren und Messzeit zusammen. Ein Schritt die Kosten zu senken ist diese Gesamtzeit zu minimieren. Wenn der Experimentator also schon vor seiner Anreise an seinem Experiment eingeführt wird, so verkürzt sich natürlich auch die Einführungsphase direkt am Experiment.

1.3 Das Hahn-Meitner-Institut Berlin GmbH als Auftraggeber

Das HMI hat dem Autor die Aufgabe übergeben, ein solches E-Learning-Tool für die Experimente der E-Halle zu entwickeln. Der Benutzer dieses Tools soll durch eine einfache Oberfläche geleitet, die Experimente erkunden und kennen lernen können. Zugleich soll er sich über die theoretischen Grundlagen, die dem Experiment als Voraussetzung dienen, informieren können.

[2] Quelle: Hahn-Meitner-Institut Berlin GmbH – www.hmi.de (Stand: 07.12.2004)

Als Informationsmedien, die Grundlage des Tools sind, kommen Bilder, Texte, Audio- und Videodaten zur Geltung.

1.4 Aufgabenstellung

Erstellt werden soll ein E-Learning-Tool, welches leicht erweiterbar und wartbar sein soll. Es soll zur Präsentation von Inhalten, die von den Experimentbetreuern und Wissenschaftlern gestellt werden, auf CD-ROM und im Internet dienen.

Es soll zwei Hauptpunkte geben, die Einweisung und die dazugehörige Theorie. Diese beiden Punkte sollen untereinander kontextsensitiv verlinkt sein. Viel Wert wird dabei auf die leichte Erweiterbarkeit der Inhalte gelegt, diese sollen dann dynamisch in einer Art Content-Management-System[3] (CMS) dargestellt werden. Die Inhalte sollen also leicht zu modifizieren und erweiterbar sein.

Dabei ist darauf zu achten, dass es sich bei den Autoren der Texte und der anderen einzubindenden Medien um Wissenschaftler handelt, die zwar mit einem Computer umgehen können, sich aber zum größten Teil mit der Programmierung nicht auskennen. Bei den meisten Wissenschaftlern handelt es sich um Physiker, die unter anderem auch T_EX[4] können.

[3] Eine Software zur Verwaltung des Inhalts einer Anwendung oder einer Internetseite.
[4] Ein Textsatzsystem mit eingebauter Makrosprache. Mit T_EX können Wissenschaftliche Formeln dargestellt werden, die weit über dem Leistungsvermögen des Formeleditors bei Microsoft Word liegen.

2 Pflichtenheft

2.1 Zielbestimmung

Mit diesem E-Learning-Tool soll der Anwender eine Einführung in die Experimente der Experimentierhalle des HMI und dessen Theorie bekommen. Es werden Multimediale Komponenten mit eingebunden. Es handelt sich um ein vollkommen dynamisches System.

Soll-Kriterien

Das System muss von außen wartbar und erweiterbar sein. Medien sollen ohne weiteres in das E-Learning-Tool eingebunden werden können. Zu den Medien gehören Texte, Bilder und Videos. Die Anwendung soll klein, übersichtlich und flexibel sein, d.h. die Texte und die weiteren Medien werden dynamisch nachgeladen.

Es soll zwei Bereiche geben, den Einweisungsbereich und den Theoriebereich. Die Bereiche lassen sich durch zwei Menüpunkte trennen. Des Weiteren soll ein Glossar angeboten werden, in dem wichtige Begriffe erklärt werden.

Die Inhalte der Einweisung und die der Theorie müssen sich auf die gleiche Weise darstellen lassen. Menüunterpunkte werden dynamisch aus den Inhalten generiert.

Die Texte können unter anderem griechische Buchstaben enthalten. Diese sollen auch auf dem Bildschirm dargestellt werden.

Von jedem Punkt aus der Anwendung kann zu jeder Zeit ein anderer Punkt aufgerufen werden.

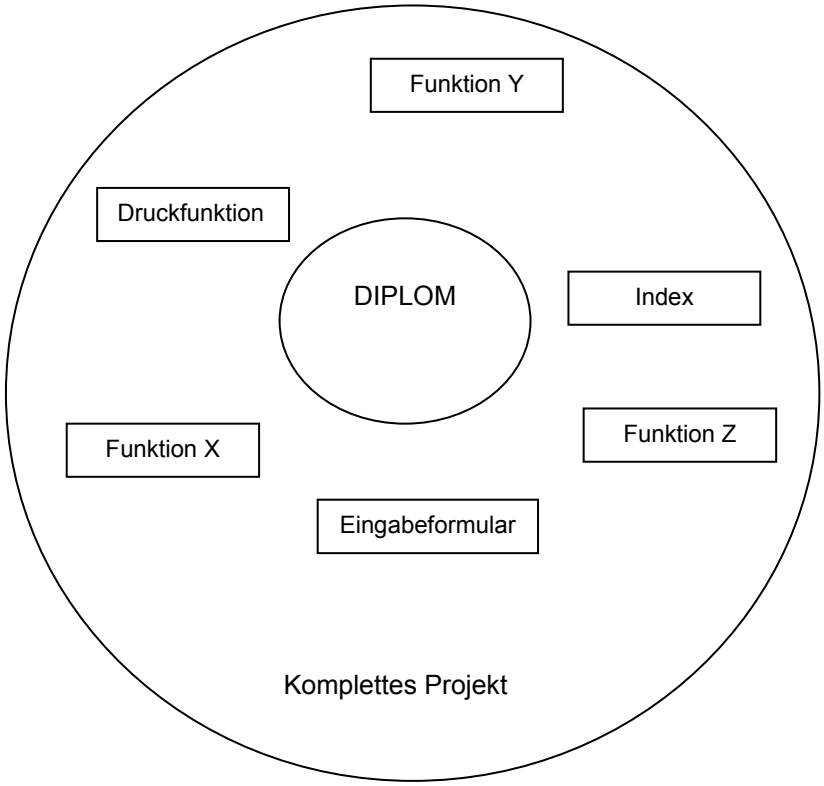


Abbildung 2.1.1: Abgrenzung - Fertige Anwendung und Diplom

In Abbildung 2.1.1 ist zu sehen, dass die hier vorgestellte Anwendung nur ein Teil eines umfangreicheren Projektes ist. Aufgrund der kurzen Zeit musste abgeschätzt werden, welche Teile des Projekts später fertig gestellt werden und welche in dieser Zeit realisierbar sind.

Wunschkriterien

Die Anwendung soll auf lange Sicht aus dem Internet heraus aufgerufen und gewartet werden können.

Eine Druckfunktion wäre unter Umständen wünschenswert. Mit dieser könnte der Anwender sich dann Texte und/oder Bilder ausdrucken und zu einem anderen Zeitpunkt studieren, ohne am Computer zu sitzen.

Formeln, welche aus mehreren Zeilen bestehen, zum Beispiel durch ein Bruchstrich getrennte, aus dem wissenschaftlichen Bereich im Text einzugeben und darzustellen, wäre eine weitere Ergänzung.

Bilder sollen ohne Größenvorgabe eingebunden werden können, was die Arbeit bei der Pflege der Anwendung erleichtern würde.

Eine komfortable Eingabemaske für die Themen und deren dazugehörige Medien wäre für den administrierenden Nutzer eine weitere Hilfe.

Eine Datenbankanbindung ist möglich, wäre aber mit einigem Aufwand verbunden.

Abgrenzungskriterien

Das E-Learning-Tool stellt auch eine Hilfe für den Experimentgast dar und soll nicht nur als alleiniges Lerntool dienen.

Inhalte werden von den Wissenschaftlern geliefert und gehören nicht zum Umfang des Produkts. Es werden lediglich beispielhafte Dateien mitgegeben.

2.2 Produkteinsatz

Anwendungsbereiche

Alle Nutzer der Experimente in der E-Halle des HMI können dieses Produkt benutzen. Außerhalb des Instituts können die Gastexperimentatoren sich vor Ankunft mit dem Tool informieren.

Zielgruppen

Zielgruppe sind vor allem die Gäste des HMI, die in der E-Halle Experimente durchführen wollen und sich noch nicht so gut mit dessen Gerätschaften auskennen. Weiterhin kann jeder Mitarbeiter oder Gast, der an Neutronenstreuexperimenten arbeitet oder sich darüber informieren möchte, dieses Produkt nutzen.

Betriebsbedingungen

Die Anwendung kann theoretisch überall und zu jeder Zeit benutzt werden, da sie sich voraussichtlich aus dem Internet aufrufen lässt und auch auf CD-ROM benutzt werden kann. Einzig die Inhalte halten eventuell davon ab, die Anwendung außerhalb des Instituts zu benutzen, d.h. wenn Inhalte angeboten werden die nicht für die Öffentlichkeit

bestimmt sind, so wird die Anwendung wahrscheinlich nur im Intranet des HMI zu erreichen sein.

2.3 Produktumgebung

Software

Für den Betrieb im Internet wird für das Produkt ein Webserver gebraucht. Auf diesem muss ca. 200Kb Platz für die Grundinstallation der Anwendung vorhanden sein. Allein die externen Daten, die die Anwendung darstellen soll, könnten unter Umständen sehr viel Speicherplatz erfordern. Aber das kommt auf den Umfang der Daten an. Da aber, wie in den Betriebsbedingungen eben schon erwähnt, diese Daten durchaus nicht für die Öffentlichkeit bestimmt sein könnten, könnte die Anwendung in einem passwortgeschützen Bereich, zum Beispiel durch Htaccess[5], des Servers laufen. Sollte dieses der Fall sein, so kann dem Gast, der die Anwendung benutzten möchte oder soll, für eine bestimmte Zeit, ein Passwort übermittelt werden, so dass dieser das Tool benutzen kann.

Der Client sollte einen Webbrowser und einen Internetanschluss besitzen. Für den Browser wird das aktuelle Flash 7 Plug-In von Macromedia[6] benötigt.

Für die CD-ROM wird ein PC mit installiertem Microsoft Windows 2000 oder XP vorausgesetzt.

Hardware

Für die Internetanbindung empfiehlt sich mindestens ein DSL-Anschluss, ein herkömmliches 56k-Modem reicht aber auch. Da die Anwendung aber größtenteils im HMI-LAN benutzt wird, dürfte es wohl kein Geschwindigkeitsproblem geben.

Der Rechner sollte nicht weniger als 800 MHz haben, darunter könnte die Anwendung etwas träge reagieren. Dieses gilt sowohl für den Client als auch für die Workstation mit der CD-ROM.

Die Grafikausgabe erfolgt standardmäßig mit einer Auflösung von 1024x768 Punkten, lässt sich aber skalieren, wodurch es auf die Augen des Betrachters ankommt. Ist die Bildschirmauflösung kleiner, so passt sich die Ausgabe an und wird auch kleiner, ist sie größer, so wird die Ausgabe vergrößert.

Für den Audio- und Videogenuss empfehlen sich Standartlautsprecher mit einer konfigurierten Stereosoundkarte.

2.4 Produktübersicht

Die Anwendung besteht aus einer Flashdatei. Diese wird auf der Clientseite mittels einer HTML-Seite dargestellt. Wird die Anwendung von der CD-ROM gestartet, also auf einer Workstation, wird die Flashdatei über den mitgelieferten Flashplayer, einem Projektor der die Datei darstellt, angezeigt.

[5] Ein einfaches, aber effektives, Verfahren um Webinhalte zu schützen
[6] http://www.macromedia.com (Stand: 05.12.2004)

Beide Varianten arbeiten ab diesem Zeitpunkt gleich. Es werden von der Flashdatei XML-Dateien nachgeladen, die verarbeitet werden und der Flashdatei mitteilen, welche Texte und Mediendateien vorliegen.

Der Benutzer kann jetzt über ein Menü die Anwendung steuern und sich die Inhalte anzeigen lassen.

Der Administrator, der die Inhalte vorher bearbeitet und zur Verfügung stellen möchte, muss diese XML-Dateien vorher bearbeiten. Er muss auch die Mediendateien in die richtigen Verzeichnisse kopieren.

Zu einem späteren Zeitpunkt ist es dann auch möglich, die XML-Daten über eine geeignete Maske zu bearbeiten und die zugehörigen Mediendateien über diese hoch zu laden.

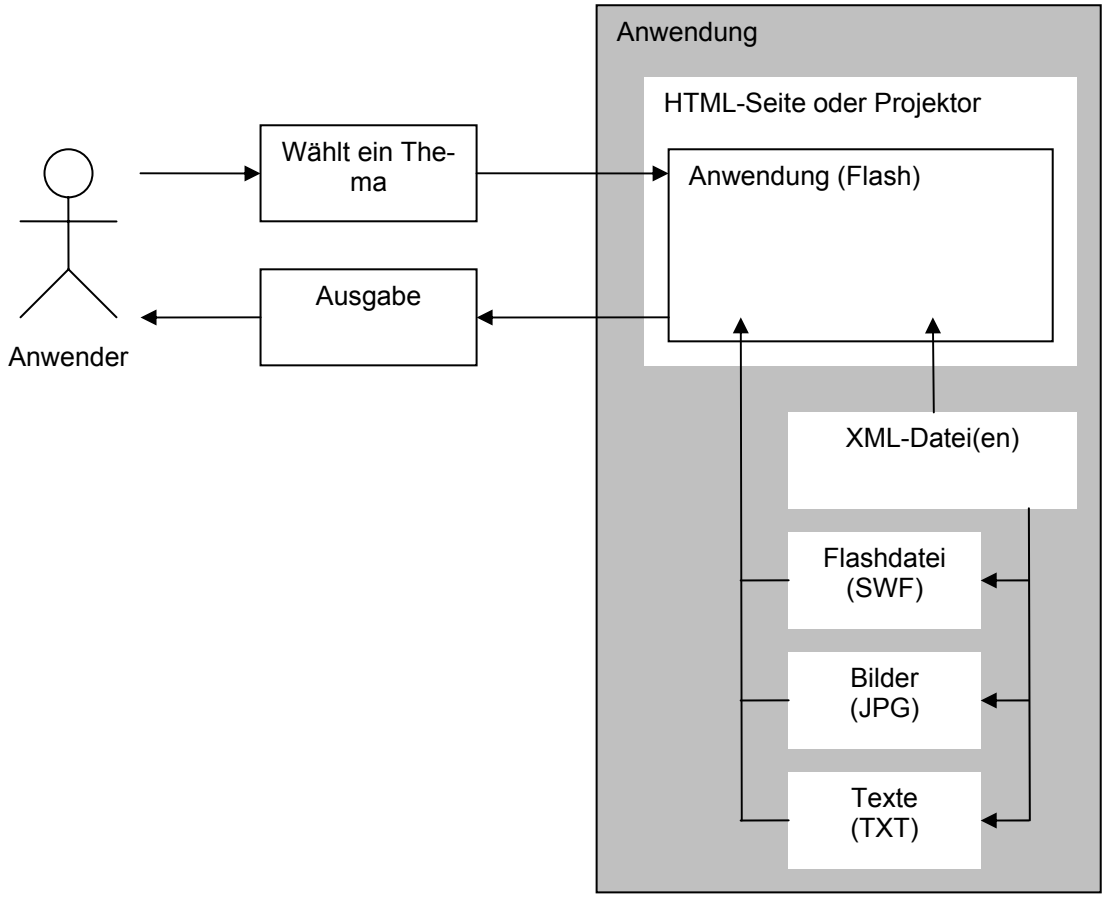


Abbildung 2.4.1: Übersichtsdiagramm

2.5 Produktfunktionen

Im Folgenden werden die Funktionen des Produkts bzw. der Anwendung grob beschrieben.

Anwender

Der Anwender kann im Prinzip die Anwendung nur bedienen, von ihm werden keine Tastatureingaben erwartet. Eingaben werden vom Benutzer nur gefordert, falls die An-

wendung auf einem Server läuft und der Bereich passwortgeschützt ist, was aber in diesem Falle nicht die Anwendung selber betrifft.

Screen	Aktion	Beschreibung
Bedienoberfläche	A1.01 Menüpunkt Auswählen	Der Benutzer wählt mit dem Mauszeiger einen Menüpunkt aus dem Menü aus.
	A1.02 Medienwahlbutton	Falls zu einem Thema mehrere Medien zur Verfügung stehen, kann der Benutzer diesen Button drücken um ein Medium weiter zu schalten.
	A1.03 Weitere Themen Auswahl	Hier kann der Benutzer einen Link auswählen, um schnell zu einem Folgethema zu gelangen.
	A1.04 Drucken	Diese Funktion wird zu einem späteren Zeitpunkt zur Verfügung stehen. Es soll das Thema samt Bild auf DIN A4 ausgedruckt werden können.
	A1.05 Textfeld scrollen	Bei einem langen Text kann der Benutzer mittels Scrollbalken den Text hoch und runter scrollen.
	A1.06 Textfeld Link	Im Textfeld werden Links dargestellt. Diese führen bei Betätigung in das Glossarfenster und zeigen den entsprechenden Eintrag.
	A1.07 Anwendung beenden (nur CD-ROM Version)	Der Anwender drückt diesen Button oder den entsprechenden Menüpunkt zum beenden der Anwendung.
	A1.08 History Vor-/Zurückblättern	Der Anwender kann die Buttons drücken, um durch kürzlich besuchte Themen zu navigieren.

Abbildung 2.5.1: Produktfunktionen auf der Bedienoberfläche

Screen	Aktion	Beschreibung
Bedienoberfläche mit Glossarfenster	A2.01 Glossarfenster schließen	Ein Button zum Schließen des Glossarfensters.
	A2.02 Glossareintrag wählen	Eine Liste mit allen Glossareinträgen ist sichtbar. Hier kann ein Eintrag gewählt werden.

Abbildung 2.5.2: Produktfunktionen im Glossar

Administrator

Der Administrator hat die Aufgabe, die Texte und Mediendateien in die Anwendung einzupflegen. Hier werden XML-Dateien bearbeitet und Mediendateien in die richtigen Verzeichnisse kopiert.

Geschäftsprozesse

Es werden hier einige ausgewählte Geschäftsprozesse dargestellt.

Geschäftsprozessbezeichnung: Anzeige Thema	
Ziel:	Anzeige des neuen Themas, d.h. Text und die dazugehörigen Mediendateien werden geladen und angezeigt.
Kategorie:	Primär
Vorbedingung:	Das Menü wurde erfolgreich aufgebaut und der Benutzer hat ein Menüpunkt aus dem Bereich Einweisung oder Theorie des Menüs ausgewählt. Des Weiteren kann in einem Bereich, in welchem weitere Themen angezeigt werden, ein Link ausgewählt werden.
Nachbedingung Erfolg:	Anzeige des gewählten Themas korrekt aufgebaut.
Beschreibung:	Der Benutzer geht mit dem Mauszeiger über das Menü und klickt einmal, damit die Unterpunkte angezeigt werden. Jetzt kann er einen Menüpunkt oder ein weiteres Untermenü anklicken, welches dann aufgeht worin er dann einen Menupunkt anwählt. Ist die linke Maustaste nun gedrückt worden, werden die dem Thema entsprechenden Mediendateien geladen und angezeigt. Weiterhin kann der Benutzer in einem Bereich, in dem weiterführende Themen angezeigt werden, einen Link auswählen.

Abbildung 2.5.3: Geschäftsprozess Anzeige Thema

Geschäftsprozessbezeichnung: Anzeige Glossarfenster	
Ziel:	Anzeige des Glossarfensters mit einem vorher definierten Thema oder aber ein Thema kann ausgesucht werden.
Kategorie:	Primär
Vorbedingung:	Der Menüpunkt Extras/Glossar wurde gewählt oder im Textfeld wurde ein Link angewählt.
Nachbedingung Erfolg:	Das Fenster Glossar wird geöffnet und die entsprechenden Inhalte werden gezeigt.
Beschreibung:	Über den Menüpunkt Extras/Glossar oder über einen Link im Textfenster lässt sich das Glossarfenster öffnen. Dieses zeigt als Titel den gewählten Glossareintrag an und den entsprechenden Eintrag auf der rechten Seite in einem Textbereich. Auf der linken Seite werden in einer Liste alle Glossareinträge, die man dort auswählen kann, angezeigt.

Abbildung 2.5.4: Geschäftsprozess Anzeige Glossarfenster

Geschäftsprozessbezeichnung: Bild/Medieninhalt weiter	
Ziel:	Durch Klick auf den entsprechenden Button wird das nächste Bild bzw. der nächste Video- oder Audioinhalt angezeigt.
Kategorie:	Sekundär
Vorbedingung:	Es existieren mehr Bilder oder Medieninhalte auf der CD oder dem Webserver zu dem angezeigten Thema.
Nachbedingung Erfolg:	Das Bild oder die Audio-/Videodatei wird angezeigt.
Beschreibung:	Der Benutzer kann sich mehrere Medieninhalte anschauen, sofern diese vorhanden sind. Es erscheinen Buttons mit denen zwischen den Inhalten hin- und hergeschaltet werden kann.

Abbildung 2.5.5: Bild/Medieninhalt weiter

Geschäftsprozessbezeichnung: Verwandtes Thema anzeigen	
Ziel:	Verwandtes Thema wird angezeigt, nachdem auf den entsprechenden Link geklickt wurde.
Kategorie:	Primär
Vorbedingung:	Ein Thema wurde gewählt und es wird unter verwandte Themen ein weiteres Thema angezeigt.
Nachbedingung Erfolg:	Das entsprechend ausgesuchte Thema wird mit allen Inhalten dargestellt.
Beschreibung:	Zu jedem Thema gibt es einen Abschnitt, bei dem verwandte Themen eingetragen werden können. Diese verwandten Themen werden bei der Anzeige in einem eigenen Bereich dargestellt. Hier kann der Benutzer jetzt ein Thema anklicken, welches sich dann öffnet.

Abbildung 2.5.6: Verwandtes Thema anzeigen

Geschäftsprozessbezeichnung: Historyfunktion Vor/Zurück	
Ziel:	Ein Thema vor- oder zurückblättern, in dem der Anwender schon war. Funktion wie im Webbrowser.
Kategorie:	Primär
Vorbedingung:	Es wurde schon mehr als ein Thema betrachtet und der Anwender hat den Zurück-Button geklickt oder er hat den Zurück-Button schon betätigt und drückt jetzt den Vor-Button.
Nachbedingung Erfolg:	Das gewünschte Thema wird angezeigt.
Beschreibung:	Diese Funktion arbeitet so wie die Historyfunktion in einem Browser, zum Beispiel dem Internet Explorer. Ein schon einmal besuchtes Thema kann per Zurück-Button wieder aufgerufen werden. Hat der Benutzer diesen Button schon betätigt, so kann er auch wieder mit dem Vor-Button durch die Themen vorwärts navigieren.

Abbildung 2.5.7:Historyfunktion Vor/Zurück

Geschäftsprozesse in der weiterentwickelten Version

Hier noch eine kleine Auswahl an Geschäftsprozessen, die in einer späteren Version der Anwendung umgesetzt werden sollen. Dieses sind Funktionen, die das Projekt nach der Abgabe dieser Arbeit hinzubekommt. Es werden höchstwahrscheinlich noch mehr Funktionen werden. Aber das kann erst nach einer längeren Testphase ermittelt werden, welche Funktionen dieses werden.

Geschäftsprozessbezeichnung: Anzeige Index	
Ziel:	Ein Index zum suchen in den Themen wird angezeigt.
Kategorie:	Sekundär
Vorbedingung:	Index wurde ausgewählt.
Nachbedingung Erfolg:	Anzeige aller im Moment verfügbarer Indexeinträge.
Beschreibung:	Hier kann der Benutzer nach Einträgen suchen oder sich aber Themen aussuchen, die er unter einem anderen Namen kennt. Die Indexeinträge sind Einträge, die zu einem Thema in der XML-Datei extra angegeben werden.

Abbildung 2.5.8: Geschäftsprozess Anzeige Index

Geschäftsprozessbezeichnung: Aktuelles Thema drucken	
Ziel:	Das aktuelle Thema, welches gerade angezeigt wird, wird auf DIN A4 Papier ausgedruckt. Ausgedruckt wird der Text samt Bild(er).
Kategorie:	Sekundär
Vorbedingung:	Ein Thema wurde gewählt und Druckbutton wurde gedrückt.
Nachbedingung Erfolg:	Das entsprechende Thema wird mit allen Inhalten ausgedruckt. Voraussetzung ist ein installierter Drucker.
Beschreibung:	Wenn der Benutzer ein Thema angewählt hat, es aber zur weiteren Verwendung ausdrucken möchte, so kann er es über diesen Button.

Abbildung 2.5.9: Aktuelles Thema ausdrucken

2.6 Produktdaten

Aus Benutzersicht fallen bei der Anwendung keine Daten an, da Daten nur abgerufen und nicht gespeichert werden. Daten werden nur vom Administrator der Anwendung eingepflegt. Von der Anwendung selber werden keine Daten zur Speicherung produziert.

2.7 Produktleistungen

Da das Produkt vordringlich von CD-ROM gestartet wird, ist davon auszugehen, dass die Daten, die geladen werden, schnell genug angezeigt werden.

Über das Internet gestartet ist allerdings davon auszugehen, dass es sich teilweise um große Datenmengen handelt, die nachgeladen werden. Dadurch kann es zu Verzögerungen in der Anzeige kommen.

2.8 Benutzeroberfläche

Die Benutzeroberfläche ist so konzipiert, dass sie einem weitgehenden Webstandart entspricht. Es gibt also eine Navigationsleiste und klar definierte Inhaltsbereiche. So findet sich der Benutzer schnell zurecht.

Das E-Learning-Tool setzt eine Auflösung von 1024x768 Punkten voraus. Ist die voreingestellte Auflösung des jeweiligen Systems allerdings kleiner, so passt sich das Fenster automatisch an. Die Darstellungsqualität lässt dann aber dementsprechend nach.

Die Oberfläche ist durchweg mit der Maus zu bedienen. Alle Interaktionen erfordern keine Tastatureingabe. Die Menüführung ist ähnlich der von ganz normalen Windowsprogrammen, sieht aber etwas anders aus. Das Menü ist etwas tiefer zur Bildschirmmitte hin gelegt, da in dem oberen Bereich Platz für Logos etc. bleiben muss.

Designvorgaben des Unternehmens (HMI) können leider nicht komplett umgesetzt werden, da die Software, mit der gearbeitet wird, die Anpassung nicht in einem akzeptablen Rahmen ermöglicht. Bestimmte Komponenten der Entwicklungssoftware lassen nur die Auswahl aus drei Farbstilen zu.

2.9 Testfälle

Getestet wird das Produkt schon während der Entwicklungsphase, da die Entwicklungsumgebung viele neue Funktionen bereithält. Diese müssen erprobt und ausgetestet werden. Nur nach erfolgreichen Tests werden die Komponenten in die Anwendung übernommen.

Nach der Fertigstellung wird das Produkt gründlich durch Testpersonen getestet werden. Sind die XML-Inhalte konsistent und werden korrekt dargestellt? Werden die Dateien korrekt geladen und angezeigt? Funktionieren die implementierten Funktionen fehlerfrei?

Es könnten zum Beispiel Fragebögen entwickelt werden, in denen die Testpersonen genau beschreiben, welche Aktionen sie ausführen und zu welchem Ergebnis sie geführt haben.

2.10 Entwicklungsumgebung

Als Entwicklungsumgebung dient dem Produkt das Autoringtool Macromedia Flash MX 2004 Professional. Die Konfigurationsdateien, d.h. die XML-Dateien, werden mit dem Editor UltraEdit[7] editiert oder einem anderen Editor der UTF-8 Export beherrscht.

Video- und Audiodateien werden in einem speziellen Format, dem FLV-Format von Flash MX 2004, eingebunden.

[7] Texteditor der Firma IDM Computer Solutions, Inc. – http://www.ultraedit.com (Stand: 18-12-2004)

3 Technisches Umfeld

3.1 Warum ein Autorensystem

Autorensysteme sind Anwendungen, mit denen Applikationen nach dem Prinzip des WYSIWYG[8] erstellt werden können. Es ist also eine Art visuelle Programmierumgebung, mit deren Hilfe man die zu erstellende Applikation graphisch bearbeiten kann.

Auf dieser visuellen Entwicklungsoberfläche wird die zu erstellende Applikation nach dem Baukastenprinzip erstellt, d.h. während der Bearbeitung erscheint die entwickelte Oberfläche genauso oder ähnlich wie bei dem fertigen Produkt.

Durch die Integration von einzelnen vorgefertigten Komponenten, die in eigenen Anwendungen verwendet werden können, werden Anwendungen in einem Bruchteil der Zeit erstellt als dies mit konventioneller Programmierung möglich wäre.

Autorensysteme sind weniger auf Programmierer abgestimmt, sondern vielmehr auf Graphiker und sind zur Erstellung interaktiver Multimedia-Anwendungen bestimmt.

In Autorensystemen können Text und Multimediainhalte, wie Audio- und Videodateien, innerhalb kürzester zusammengefügt werden. Grafiken können mit Aktionen belegt werden, so dass bei Berührung mit dem Mauszeiger diese ausgeführt wird. Zum Beispiel könnte eine Melodie oder ein Film abgespielt werden, wenn der Mauszeiger über eine bestimmte Fläche kommt.

Da es sich um eine visuelle Oberfläche handelt, kann das Design schnell geändert werden falls es, wie es in Firmen öfter vorkommt, Unstimmigkeiten im Design gibt.

Da die meisten Systeme nicht genau über die Komponenten verfügen, die man gerade in diesem Augenblick braucht, beherrschen sie zudem eine Scriptsprache, mit der zum Beispiel eigene Effekte oder aber auch Komponenten erstellt werden können.

Scriptsprachen sind Programmiersprachen, in denen der Quellcode als Textdatei vorliegt. Diese Textdateien können mit einem normalen Editor bearbeitet werden. Scripte werden entweder vor der Ausführung kompiliert oder interpretiert. In den meisten Autorensystemen werden sie allerdings kompiliert und liegen bei der Ausführung als binäre Datei vor. Handelt es sich um ein Autorensystem welches einen Player benötigt, so liegt der Code als Bytecode[9] vor, d.h. erst im Player geladen wird die Anwendung kompiliert und ausgeführt.

Was für Autorensysteme gibt es also? Der Autor möchte sich hier auf zwei Autorensysteme beschränken, die er während seines Studiums kennen gelernt hat. Das sind zum einen Macromedia Flash und zum anderen Macromedia Director.

Beide Systeme sind die marktführenden Autorensysteme für Multimediaprojekte. Sie sind beide sehr umfangreich und enthalten alles was man für das schnelle Erstellen von Multimedia CD-ROMs oder Internetanwendungen benötigt. Directordateien (dcr) brau-

[8] What You See Is What You Get ("Was Du siehst, das bekommst Du auch"). Ein Dokument wird schon während der Entwicklung so am Bildschirm angezeigt, wie es bei der Ausgabe (z.B. Drucker) erscheint. Bei Anwendungen sieht man in der Entwicklungsumgebung schon die grafische Darstellung des fertigen Produkts.

[9] Bei der Kompilierung von Quelltext, z.B. von Java, entsteht nicht Maschinencode, sondern ein Code, der vor der Ausführung in einer virtuellen Maschine aufbereitet und erst dann ausgeführt wird. Der Unterschied zum Quelltext für ein Interpreter, ist das der Bytecode in der Regel Maschinenunabhängig ist.

chen zur Darstellung im Internetbrowser einen Shockwave-Player und für Flashdateien (swf) wird der Flash-Player in der aktuellen Version 7 benötigt.

Die Programme verfügen neben der umfangreichen Komponentenansammlung über Scriptsprachen. Bei Director ist es Lingo und bei Flash ActionScript, wobei ActionScript nunmehr in der Version 2 vorliegt.

Die folgende Statistik in Abbildung 3.1.1 zeigt einen Vergleich von installierten Browser-Plugins.

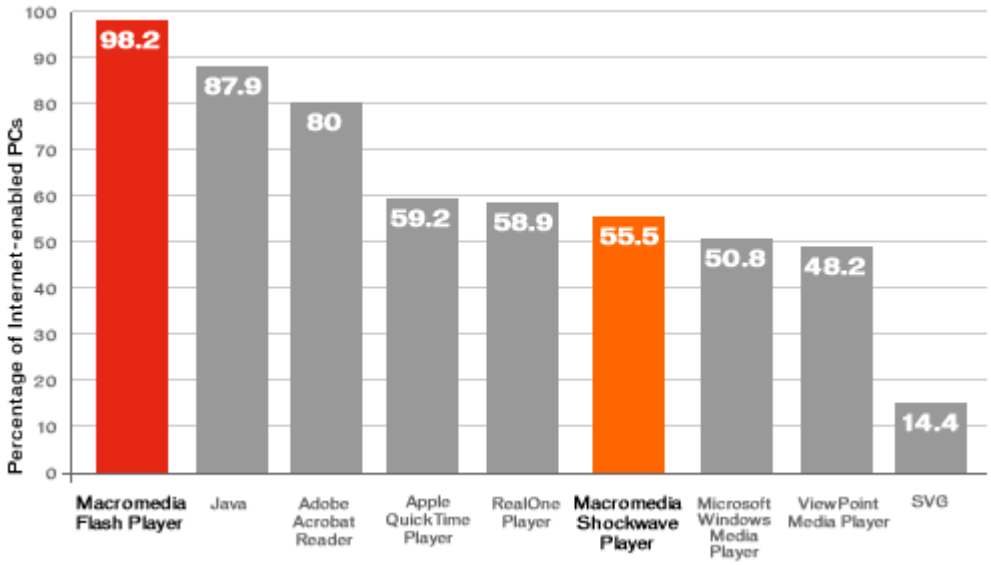


Abbildung 3.1.1: Statistik installierter Browser-Plugins[10]

Aufgrund des hohen Verbreitungsgrades des Flash-Players, der bei rund 98,2 Prozent aller Internetbenutzer installiert ist und der flexibleren Gestaltungsmöglichkeiten, wird für dieses Projekt das Autorensystem Macromedia Flash MX 2004 Professional verwendet. Außerdem kennt sich der Autor mit Flash besser aus und da der strenge Zeitrahmen für dieses Projekt es nicht erlaubt, sich in einer Anwendung von vornherein einzuarbeiten, ist diese Entscheidung nur zu befürworten.

Zudem gibt es gegenüber dem Shockwave-Player Performancevorteile zugunsten des Flash-Players, d.h. die Anwendungen sind kleiner, was wiederum für die gute Komprimierung von Flash-Dateien spricht und sie sind dadurch natürlich deutlich schneller geladen.

Im Gegensatz zu vielen anderen Meinungen ist Macromedias Flash sehr wohl und auch sehr gut für Präsentationszwecke geeignet. Es lassen sich hervorragende Multimedia CD-ROMs produzieren. Gerade Flash MX 2004 Professional ist auch dafür geeignet, wie mit Microsofts Powerpoint, Folienpräsentationen zu erstellen. Dies lohnt sich aber nur wenn Flash MX 2004 Professional schon vorhanden ist und nicht extra für Folienpräsentationen angeschafft wird, da Flash ein wenig zu teuer wäre nur für diesen Einsatz.

[10] Quelle: http://www.macromedia.com/software/player_census/flashplayer/ Stand: 14.12.2004

3.2 Flash

Macromedia Flash MX 2004 Professional gibt es für zwei Betriebssysteme, für Mac OS oder aber Windows. Voraussetzung ist ein leistungsfähiger Rechner mit mindestens 800 MHz, um ein angenehmes Arbeiten zu gewährleisten. Minimale Betriebssystemversion für Macintosh-Benutzer ist das OS 10.2.6 oder OS X, für Windows-Anwender Windows 98SE. Dieses Projekt wird aber unter Windows entwickelt. Für die Online-Version ist dies aber unerheblich, da nur das Browser-Plugin benötigt wird, welches ja für mehrere Betriebssysteme und dessen Browser verfügbar ist.

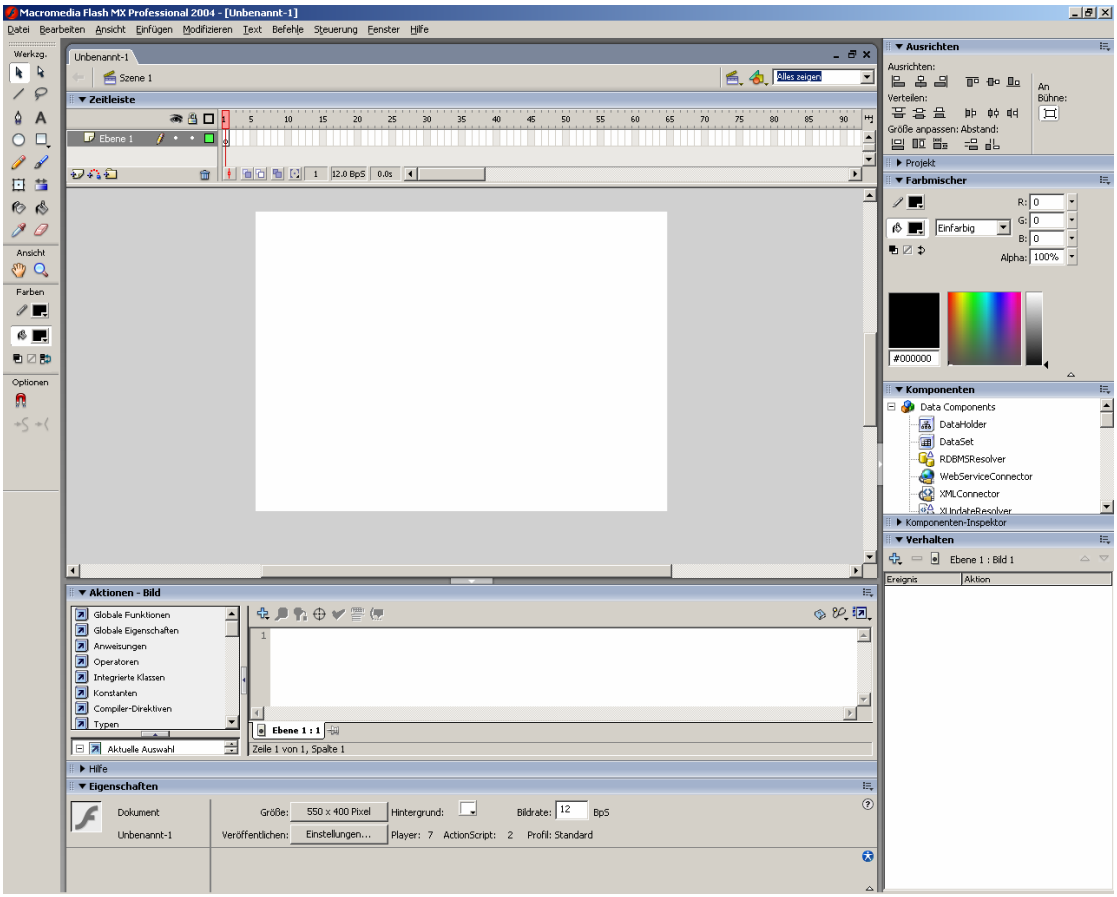


Abbildung 3.2.1: Die Oberfläche von Macromedia Flash MX 2004 Professional

Flash MX 2004 (Professional) und auch schon vorhergehende Versionen von diesem Produkt, basiert auf dem Konzept mit einer Zeitleiste. Diese ist in Abbildung 3.2.1 im oberen Drittel des Bildschirms zu sehen. Alle Funktionen werden zeitabhängig gesteuert. In dieser Zeitleiste, die nicht in Minuten aufgeteilt ist, sondern in Frames, also Einzelbildern, kann der Anwender zeichnen oder auch einzelne Bilder einfügen. Mit der Zeitleiste wird der Ablauf des Films gesteuert, aber auch wann welche Scripte zum Einsatz kommen.

In der Mitte des Bildschirms ist die Bühne zu sehen. Auf ihr kann der Anwender mit den Werkzeugen, die auf der linken Seite angeordnet sind, zeichnen. Auf der rechten Seite neben der Bühne sind Bedienfelder zu sehen. In ihnen befinden sich zum Beispiel einzelne Funktionen wie der Farbmischer, die Komponenten oder aber auch die Bibliothek, in der die einzelnen Objekte und Komponenten eines Flashfilms wieder zu finden sind. Die Bedienfelder sind auf- und zu klappbar und sie können zu einzelnen kleinen Fenstern angeordnet werden, die so frei auf dem Bildschirm platzierbar sind.

Unter der Bühne befinden sich noch zwei der wichtigsten Fenster. Das Fenster Aktionen, in dem die Scriptsprache ActionScript angewendet wird und das Fenster Eigenschaften. In diesem werden alle Eigenschaften des jeweilig angeklickten Objekts angezeigt und können verändert werden.

In der Zeitleiste kann natürlich nicht nur eine Zeitebene verwendet werden, sondern es können mehrere Ebenen, die so verwendet werden wie zum Beilspiel in Adobe Photoshop, angelegt werden.

Der versierte Anwender erstellt Movieclips und platziert diese in den einzelnen Frames der Zeitleiste. Movieclips sind praktisch kleine Filme im Flashfilm, die eine eigene Zeitleiste haben und dort beliebig platziert werden können. Jeder Movieclip ist eine eigene Instanz, also ein eigenständiges Objekt. Ähnlich wie der Umgang mit Klassen in Java, können die dem Objekt zugehörigen Variablen verändert werden und so das Verhalten des Objekts beeinflusst werden. Movieclips können verschachtelt werden, d.h. ein Movieclip kann wiederum ein Movieclip enthalten usw.

In Flash können Komponenten, wie zum Beispiel der XML-Connector verwendet werden. Diese Komponenten erleichtern dem Entwickler das Leben ungemein, da Schnittstellen nach außen gelegt werden. Der XML-Connector wird dazu benutzt XML-Dateien in Flash einzubinden. Der Entwickler ist allerdings auch sehr eingeschränkt, da mit der XML-Connector-Komponente nur eine XML-Datei eingelesen werden kann und die Komponente schlecht oder gar nicht von außen beeinflusst werden kann.

Für diese Fälle gibt es vordefinierte Klassen, die in ActionScript benutzt werden können. ActionScript liegt derweil in der Version 2.0 vor und hat einen sehr großen Befehlsumfang. Um beispielsweise XML-Dateien zu handhaben, gibt es die XML-Klasse. Mit dieser kann der Entwickler in ActionScript auch verschiedene XML-Dateien handhaben und auch wesentlicher flexibler mit ihnen umgehen.

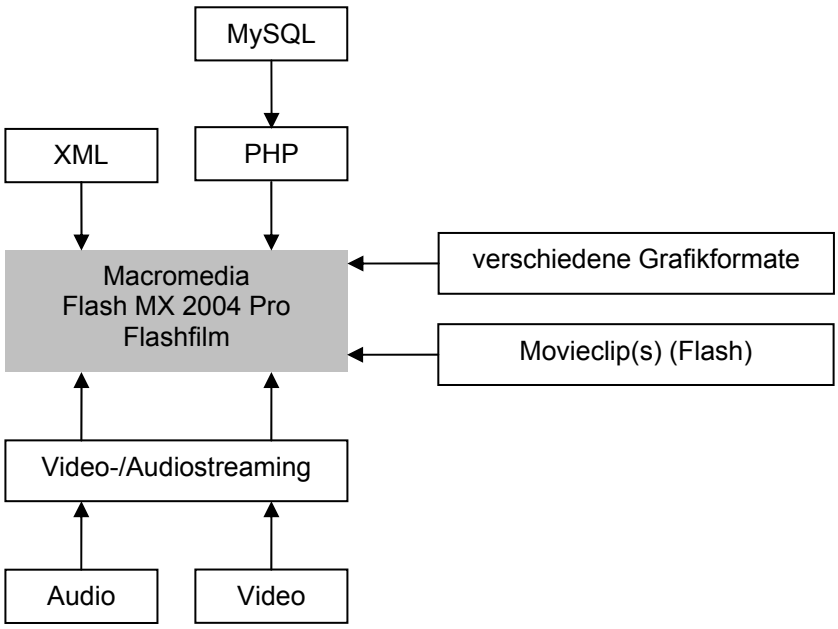


Abbildung 3.2.2: Einbindung einiger Dateien

In Abbildung 3.2.2 ist zu sehen, welche Möglichkeiten Flash zur Einbindung externer Daten bietet.

Zu beachten ist es, dass bestimmte Komponenten und Programmteile, die für die Entwicklung unter Umständen sehr wichtig sind, nur in der Professional Version von Flash MX 2004 verfügbar sind. Zu diesen gehören die Möglichkeit, mit Flash einfach Folienpräsentationen wie in Microsoft Powerpoint und Formularanwendungen zu erstellen. Einige Komponenten wie die Alert-Komponente, mit der es auf einfachstem Wege möglich ist, ein Meldungsfenster auszugeben, welches mit OK oder Abbrechen zu bestätigen ist, fehlen in der "normalen" Version ebenso wie die für dieses Projekt so wichtige MenuBar-Komponente.

3.3 Client-Server-Modell

Wie schon angesprochen läuft die Anwendung von CD-ROM. Sie soll aber auch im Internet bzw. im Intranet angeboten werden. Weswegen hier noch einmal etwas genauer auf das Client-Server-Modell eingegangen wird.

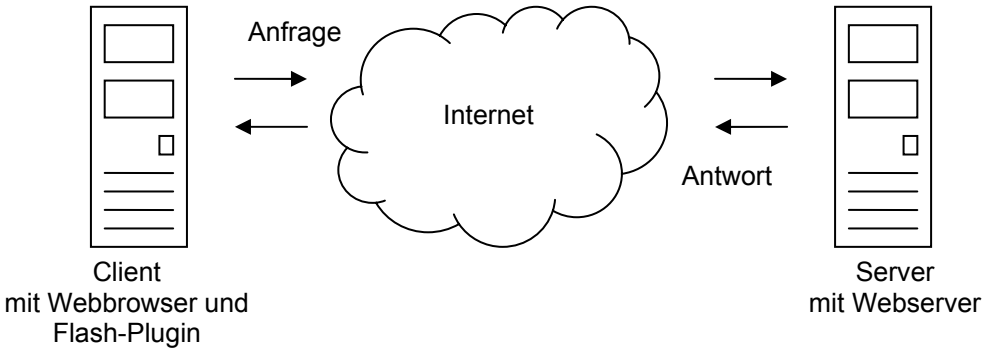


Abbildung 3.3.1: Client-Server-Modell

Der Benutzer des Clients ruft eine beliebige Webseite, über den Webbrowser, aus dem Internet auf. Der Client schickt eine Anfrage an den Server, auf dem ein Webserver installiert ist, der daraufhin antwortet und die gewünschten Informationen zur Verfügung stellt (siehe Abbildung 3.4.1). Client und Server kommunizieren über HTTP[11], der "Sprache" die im World Wide Web (WWW) Standart ist.

Die nun folgende Abbildung 3.4.2 ist genauer auf das Zusammenspiel der Anwendung mit dem Flash-Plugin ausgerichtet. Es wird davon ausgegangen, dass der Anwender vor einem Computer mit Internetzugang mit einem Browser und installiertem Flash-Plugin sitzt.

Der Webbrowser fordert eine HTML-Seite, in der die Flashanwendung eingebettet ist. Der Webserver liefert die HTML-Seite und die dazugehörige Flashanwendung, die im Flash-Plugin ausgeführt wird. Der Flash-Film, also die Anwendung, fordert jetzt die XML-Dateien an, die vom Server geliefert werden. Je nach Aktion des Benutzers werden Textdateien, Bilder oder Videos angefordert, die nach Empfang vom Server in den Flash-Film integriert werden.

[11] HyperText Transfer Protokoll

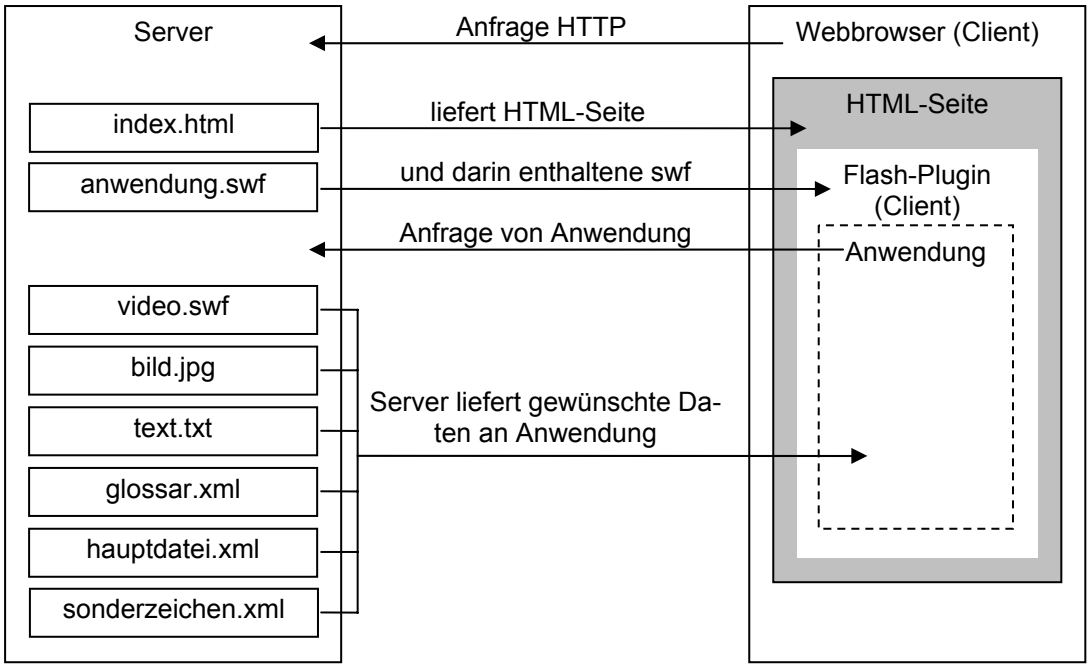


Abbildung 3.3.2: Server mit Webbrowser und Flashanwendung

3.4 XML

Die Extensible Markup Language (XML) ist eine vom World Wide Web Konsortium (W3C)[12] 1996 vorgestellte Metasprache. Sie ist eine Neudefinition einer schon länger existierenden Metasprache namens Standard Generalized Markup Language (SGML). SGML sollte auf internationalem Standart Auszeichnungssprachen definieren. Da SGML aber recht kompliziert ist, konnte und hat sich der Standart nicht durchgesetzt.

Also entwickelte das W3C einen Abkömmling von SGML und stellte im November 1996 die erste formale Definition von XML in der Version 0.01 der Öffentlichkeit vor. Erst 1998, nach einiger Überarbeitung und viel Feedback, wurde die Version 1.0 freigegeben.

XML dient dem Austausch von Daten zwischen unterschiedlichen Applikationen. Versuche XML für die Programmierung von Webseiten zu benutzen scheiterten durch die schlechte Unterstützung der Webbrowser, was bis heute der Fall ist. Die meisten Webbrowser können XML-Dokumente nur als Source, also nur den reinen Text, darstellen. Es gibt einige Ausnahmen, wie den Microsoft Internet Explorer ab der Version 5.5 und folgende, die auch mit DTDs[13] und Stylesheets umgehen können und so auch zur formatierten Darstellung, wie HTML-Seiten, genutzt werden können. Mehr dazu nachfolgend.

Der Aufbau eines XML-Dokuments ist denkbar einfach, wenn man bedenkt welche Möglichkeiten sich dahinter verbergen.

[12] World Wide Web Consortium – http://www.w3c.org (Stand: 18.12.2004)
[13] Document Type Definition

```
<?xml version="1.0" encoding="UTF-8"?>
 <glossar>
       <eintrag>
             <Schlagwort>Computer</Schlagwort>
             <text>Ein Computer (englisch compute)</text>
             <leer/>
       </eintrag>
 </glossar>
```

Listing 3.4.1: Einfaches XML-Dokument

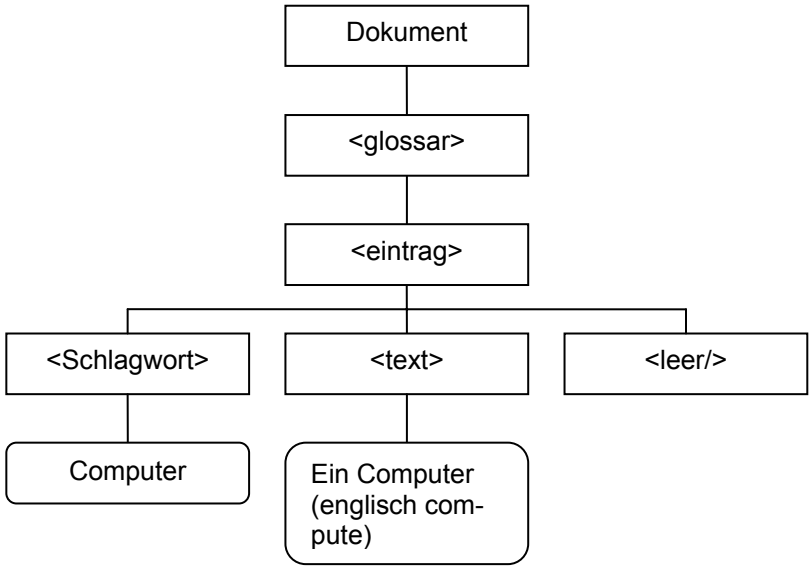


Abbildung 3.4.1: Knotenmodell des XML-Dokuments aus Listing 3.5.1

Wie in dem Listing 3.4.1 zu sehen ist, hat ein XML-Source eine gewisse Ähnlichkeit mit HTML. Man kann die Tags bei XML allerdings selber definieren und einzelne Tags, wie zum Beispiel das `<br>` Tag bei HTML, gibt es bei XML nicht. Jedes Tag muss auch wieder geschlossen werden. Kommen leere Tags vor, also welche ohne Inhalt, so kommt am Ende des Tags, bevor es geschlossen wird, ein Slash (in diesem Beispiel das Tag `<leer/>`).

Enthält ein Element weitere Elemente, so müssen die Tags in umgekehrt analoger Reihenfolge wieder geschlossen werden.

In Abbildung 3.4.1 ist das zugehörige Knotenmodel zu Listing 3.4.1 abgebildet. Dort ist zu sehen, dass die XML-Struktur sehr hierarchisch aufgebaut ist und ähnelt einer Baumstruktur.

Ein XML-Dokument muss mit einer XML-Deklaration beginnen. Diese muss immer in der ersten Zeile des Dokuments stehen. Als ISO-Normierung kommt in diesem Falle (Listing 3.4.1) UTF-8, ein Internationaler Zeichensatz mit 8 Bit, zum Einsatz. Dieser wurde gewählt da Flash MX 2004 standardmäßig mit UTF-8 kodierte Dateien umgeht.

UTF-8 steht für "8 Bit Unicode Transformation Format" und ist eine Kodierung für Unicodezeichen. Dieser universelle Zeichensatz unterstützt die meisten Alphabete und Schriftzeichen, die weltweit genutzt werden.

In der XML-Spezifikation kann man natürlich noch andere ISO-Normierungen benutzen. Diese kann man auf der Seite des W3C nachlesen. Dort kann auch die komplette Spezifikation eingesehen werden.

Normalerweise werden XML-Dokumente mit einer Document Type Definition (DTD) strukturiert. Diese DTD ist für die Logik und Struktur des Dokuments zuständig. In einer DTD wird im Grunde die Syntax des XML-Sources definiert.

Es gibt interne und externe DTDs. Die externen sollten benutzt werden, wenn es längere Dokumente werden oder aber auch wenn mehrere Dokumente dieselbe DTD benutzen, so kann sich der Entwickler auch eine Menge Arbeit ersparen.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE glossar [
    <!ELEMENT glossar (eintrag+)>
    <!ELEMENT eintrag (Schlagwort, text)>
       <!ELEMENT   Schlagwort   (#PCDATA)>
       <!ELEMENT   text         (#PCDATA)>
]>
<glossar>
    <eintrag>
       <Schlagwort>Computer</Schlagwort>
       <text>Ein Computer (englisch compute)</text>
    </eintrag>
</glossar>
```

Listing 3.4.2: XML-Datei mit interner DTD

In Listing 3.4.2 ist eine XML-Datei mit interner DTD zu sehen. In der DTD wird definiert, welche Elemente ein Dokument enthalten darf und von welchem Typ sie sind. Mit dem Eintrag `#PCDATA` wird definiert, dass das Element nur geparste Daten enthalten darf und keine weiteren Child-Elemente.

Der Ausschnitt `<!ELEMENT glossar (eintrag+)>` wird als Inhaltsmodell bezeichnet. Hier steht die Reihenfolge, in der die Elemente verwendet werden müssen. Es gibt hier nur das eine Element `eintrag`, wonach aber ein "+" folgt. Das heißt, dieses Element muss einmal vorkommen, kann aber auch öfter wiederholt werden. In diesem Eintrag sind weitere Einträge enthalten. `Schlagwort` und `text` dürfen jeweils nur einmal in `eintrag` vorkommen.

Auf die formatierte Ausgabe mit CSS[14] und XSL[15] wird hier nicht näher eingegangen. Mit diesen Formatierungssprachen ist es möglich, den XML-Dokumenten eine ansprechende Form zu geben, wie es mit HTML auch möglich ist.

Für Flash MX 2004 (Professional) sind DTDs und Stylesheets nicht ausschlaggebend. Es gibt nur wenige Funktionen mit der DTDs verarbeitet werden können. Deshalb wird

[14] Cascading Stylesheets
[15] Extensible Stylesheet Language

hier auf die Spezifikation von XML des W3C verwiesen, da für das E-Learning-Tool keine DTDs und Stylesheets benutzt werden.

3.5 Flash und XML

Um mit XML-Dokumenten in Flash zu arbeiten, hat Macromedia Flash MX 2004 (Professional) eine fertig ausgestattete Klasse mit dem Namen XML integriert. Die Methoden dieser Klasse erlauben die Arbeit mit XML-Dateien. Hier die verwendeten XML-Eigenschaften für dieses Projekt:

Eigenschaft	*Erklärung*
xmlObj.attributes	*Gibt ein Array mit den Attributen des XML-Objekts zurück. Mit xmlObj.attributes.*attributname *kann ein spezielles Attribut abgefragt werden.*
xmlObj.childNodes	*Gibt ein Array mit allen Child-Nodes eines XML-Objekts zurück.*
xmlObj.firstChild	*Liefert den ersten Child-Knoten*
xmlObj.nodeValue	*Der Wert des Knotens wird geliefert.*

Abbildung 3.5.1: XML-Eigenschaften die im Projekt verwendet werden

Eine Übersicht aller verwendbaren Methoden und Eigenschaften erhält man über die integrierte Hilfe von Flash MX 2004 (Professional).

Um ein XML-Objekt in Flash mit der XML-Datei aus Listing 3.4.1 zu erzeugen, geht man wie folgt vor. Angenommen wird hier, dass das Listing 3.4.1 als Textdatei unter dem Namen "xmlfile.xml" gespeichert wird. In dem folgenden Listing 3.5.1 wird der minimale ActionScript-Code zum einlesen der Datei in Flash gezeigt.

```
var myXML = new XML();
myXML.onLoad = istGeladen;
myXML.ignoreWhite = true;
myXML.load("xmlfile.xml");

function istGeladen(ok) {
     if(ok) {
          trace("XML-Dokument erfolgreich geladen");
     }
}
```

Listing 3.5.1: Einlesen einer XML-Datei

Mit `var myXML = new XML();` wird ein neues Objekt erzeugt. Mit diesem können alle XML-Bstandteile verwaltet und bearbeitet werden. In der nächsten Zeile wird eine Ereignisbehandlungsroutine vorbereitet, die durch das `onLoad`-Ereignis aufgerufen wird. In der dritten Zeile wird die Eigenschaft `ignoreWhite` auf `true` gesetzt. Hiermit werden leere Knoten, wie das Tag `<leer/>` nicht mit geparst. Der eigentliche Ladevorgang wird in der nächsten Zeile gestartet. Mit `myXML.load("xmlfile.xml");` wird das File in den Speicher gelesen.

Die Funktion `istGeladen(ok)` ist eine einfache Fehlerbehandlung, die bei erfolgreichem Laden eine Meldung mittels `trace` ausgibt. Wenn allerdings irgendein Problem auftritt, passiert in diesem Beispiel nichts.

Jetzt können die Methoden, die Flash zur Verarbeitung von XML-Objekten bietet, angewandt werden.

Hier einige Beispiele um an die Daten bzw. Knoten zu kommen

```
trace(myXML.firstChild.nodeName.toString());
trace(myXML.firstChild.childNodes[a].childNodes[b].firstChild.nodeValue.toStri
ng());
trace(myXML.firstChild.childNodes.length);
```

Listing 3.5.2: Beispiele für das Traversieren eines XML-Stranges

Mit der ersten Zeile aus Listing 3.5.2 wird mit dem XML-Objekt aus Listing 3.5.1 "glossar" ausgegeben. Interessanter wird es in der zweiten Zeile. Hier könnte man mehrere Einträge, sofern vorhanden, durchlaufen. Mit `childNodes[a]` durchläuft man, wenn für a Werte angegeben werden, die jeweiligen Einträge die unter `<eintrag>…</eintrag>` angegeben sind. Wenn für `a` der Wert 0 gegeben ist und für `b` der Wert 1, dann wird der Text "Ein Computer (englisch compute)" ausgegeben.

Das dritte trace hingegen gibt die Anzahl der Einträge zurück, in diesem Fall 1, da es nur einen Eintrag gibt.

3.6 TEX

Da bei der Erstellung von wissenschaftlichen Texten am HMI auch Sonderzeichen, wie griechische Buchstaben, benötigt werden, muss es eine Möglichkeit geben diese mit einzubinden. In der Anwendung werden die darzustellenden Texte zu einem Thema nachgeladen. Um zu verhindern, dass der Autor der jeweiligen Texte mit der Zeichentabelle arbeiten muss und so zwischen den Programmen hin- und herschalten müsste, wird die Notation der Sonderzeichen im TEX-Stil integriert.

Vorerst werden aber nur die griechischen Sonderzeichen angeboten. Da die dafür vorgesehene XML-Datei aber erweiterbar ist, lassen sich auch problemlos andere Zeichen definieren.

TEX ist ein Satzprogramm welches Mitte der 70er Jahre von Donald E. Knuth angefangen wurde zu entwickeln. Mit dem Satzprogramm TEX können nahezu alle Aufgaben gelöst werden, die dem Beruf des Setzers vorbehalten waren. Komplexe mathematische Formeln lassen sich ebenso darstellen wie umfangreiche Tabellen.

T_EX kennt ca. 900 Befehle. Davon sind etwa 300 Basisbefehle und etwa 600 Makrobefehle. Zudem sollte der Anwender von T_EX für eine erfolgreiche Anwendung sehr gute Fachkenntnisse in Programmiertechniken, Satzdruck und grafischer Kreativität besitzen.

Hier ein kleines T_EX Beispiel.

```
\documentclass[12pt,a4paper]{article}
\usepackage[T1]{fontenc}
\usepackage[latin1]{inputenc}
\usepackage{german}

\date{12. Dezember 2004}
\author{Andreas Tomm}
\title{Ein Testdokument}

\begin{document}
 \maketitle
 \newpage
 \tableofcontents
 \newpage
 \section{Erstes}
 \subsection{Unterpunkt 1}
 Hier steht der Text - äöü
 \subsection{Unterpunkt 2}
 \subsection{Unterpunkt 3}
 \section{Kapitel 2}
 \subsection{Zweites}
\end{document}
```

Listing 3.6.1: Ein T_EX Beispiel

Nach der Kompilierung, denn T_EX-Dokumente müssen für die Anzeige kompiliert werden, zeigt das Listing 3.6.1 ein Dokument mit einem Titelblatt, einer Seite mit einem Inhaltsverzeichnis und einer weiteren Seite mit dem Inhalt von zwei Kapiteln an.

Die erste Zeile klärt, welches Dokumentenformat benutzt wird und welcher Schriftgrad genutzt wird. In den nächsten drei Zeilen, in welchen `\usepackage` steht, wird geklärt, dass in diesem Dokument deutsche Umlaute ganz normal benutzt werden können, da sonst Umlaute bei der Eingabe im Text mit Anführungszeichen, gefolgt von dem entsprechenden Vokalen, also a, u oder o, eingegeben werden müssen.

Als nächstes folgt der Eintrag für das Titelblatt, das Datum der Erstellung, welches gesetzt werden kann, den Autor und den Titeltext.

Zwischen `\begin` und `\end` folgt die eigentliche Erstellung des Dokuments. Der Befehl `\maketitle` erzeugt das Titelblatt mit den oben genannten Inhalten. Mit `\newpage` wird ein Seitenumbruch gesetzt. Danach wird das Inhaltsverzeichnis mit `\tableofcontents` erstellt und wieder eine neue Seite erstellt.

Ein Kapitel wird mit `\section` gesetzt, Unterkapitel mit `\subsection`. In den geschweiften Klammern werden die Titelnamen eingetragen.

Der Inhalt, also der eigentliche Text, kann einfach ohne Befehl eingetragen werden.

3.7 Wahl des Texteditors

Für die XML-Dateien, die später von dem Administrator der Anwendung verändert werden, wird ein Texteditor gebraucht. Dieser muss UTF-8 kodierte Dateien lesen, schreiben und darstellen können. Die Symbole des griechischen Alphabets in der XML-Datei mit den TEX-Elementen sind sonst nicht erkenn- und veränderbar.

Der Editor unter Windows XP SP1, also das Notepad, kann das UTF-8 Format speichern. Empfohlen wird allerdings ein Editor mit Syntax Highlighting, d.h. die Textdarstellung wird programmiersprachenspezifisch farblich verändert (siehe Abbildung 3.7.1).

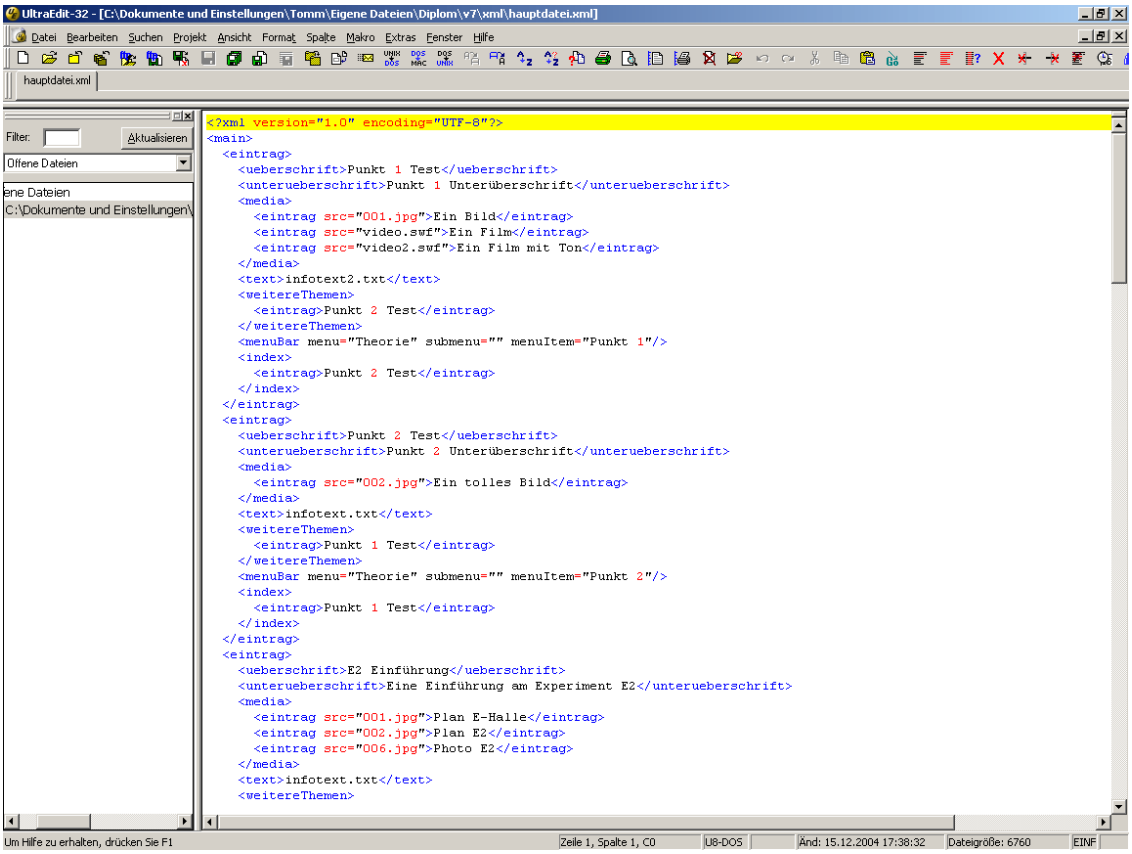


Abbildung 3.7.1: UltraEdit V10.20b – Texteditor mit Syntax Highlighting

In der Abbildung ist der Texteditor UltraEdit zu sehen, der an dieser Stelle zu empfehlen ist.

Der Editor von Macromedia Flash MX 2004 Professional, in dem die ActionScript-Dateien editiert werden, kann ebenfalls XML-Dateien lesen und schreiben. Die geringsten Kosten entstehen allerdings, wenn ein Texteditor benutzt wird, der kostenfrei von dessen Autor im Internet zur Verfügung gestellt wird.

Die Wahl des Texteditors Geschmackssache und wird dem jeweiligen Administrator der Anwendung überlassen.

4 Idee und Entwurf

Die Idee war geboren als der Autor gesehen hat, wie die Gastexperimentatoren zum HMI kamen und mit ihrer Probe, die sie ausmessen sollten, da standen. Erst einmal musste es natürlich eine grundlegende Einführung am Experiment geben. Nach einiger Zeit und etlichen Fragen später konnte es dann losgehen, der Gast durfte anfangen. Es ist natürlich klar, dass er jetzt immer wieder zum Experimentverantwortlichen kam, der ja nichts Besseres zu tun hat, und ihn mit Fragen löcherte. Manchmal hingegen ist der Experimentverantwortliche aber auch gar nicht zu erreichen.

Was wäre es schön, wenn es jetzt eine vernünftige Dokumentation mit allen Eigenheiten des Experiments geben würde. Da sich aber der Experimentaufbau während eines Jahres des Öfteren ändert, kann man keine Dokumentation drucken lassen, da dies aufgrund des mehrmaligen Nachdrucks sehr kostspielig werden würde.

Eine computergestützte Einführung mit den theoretischen Grundlagen wäre also sehr von Nützen und würde eine Menge Zeit, Kosten und Nerven sparen.

Bevor mit dem Projekt angefangen wird, muss zuerst geklärt werden, welche Inhalte und welche Datenstrukturen benutzt werden sollen. Ziel ist es, dem Benutzer die gewünschten Informationen möglichst schnell zur Verfügung zu stellen. Er soll sich nicht durch eine komplizierte Oberfläche arbeiten müssen, sondern durch wenige Klicks, an der richtigen Stelle, schnell Ergebnisse angezeigt bekommen.

4.1 Datenstrukturen und Inhalte

Interner Ablauf

Für den internen Ablauf (siehe Abbildung 4.1.1) werden XML-Dateien für Konfiguration und Inhalte benutzt. Die Dateien müssen im UTF-8 Format gespeichert werden, da Flash als Standardeinstellung dieses Format benutzt. Das Format kann aber leider nicht jeder Editor erstellen, weswegen hier UltraEdit empfohlen wird oder ein entsprechender Editor, der es kann.

Drei XML-Dateien werden von der Anwendung nachgeladen, die Haupt-XML-Datei, eine Datei für die Sonderzeichen und eine für das Glossar.

In der wichtigsten XML-Datei, der "hauptdatei.xml", werden die Inhalte für das Menü und für die Anzeige der Themen aufbereitet. In ihr steht zum Beispiel wo die Textdatei zu einem Thema zu finden ist und wo die Mediendateien zu finden sind. Diese werden dann zu dem ausgewählten Thema geladen. Mehr dazu im nächsten Kapitel.

Die Datei "sonderzeichen.xml" sagt schon vom Namen aus was sie enthält, die Übersetzung der Sonderzeichen. D.h. in dieser Datei steht das zu suchende Zeichen und das Zeichen, durch das es ersetzt werden soll.

In der Datei "glossar.xml" stehen die Schlagworte und deren Erläuterung. Wenn ein Schlagwort aus dieser Datei in dem Text zu einem Thema vorkommt, wird es automatisch verlinkt und in einer anderen Farbe dargestellt. Durch Klick auf diesen Link wird jetzt das Glossar mit diesem Eintrag angezeigt.

Diese Dateien werden von einem Script verarbeitet und für die Darstellung aufbereitet.

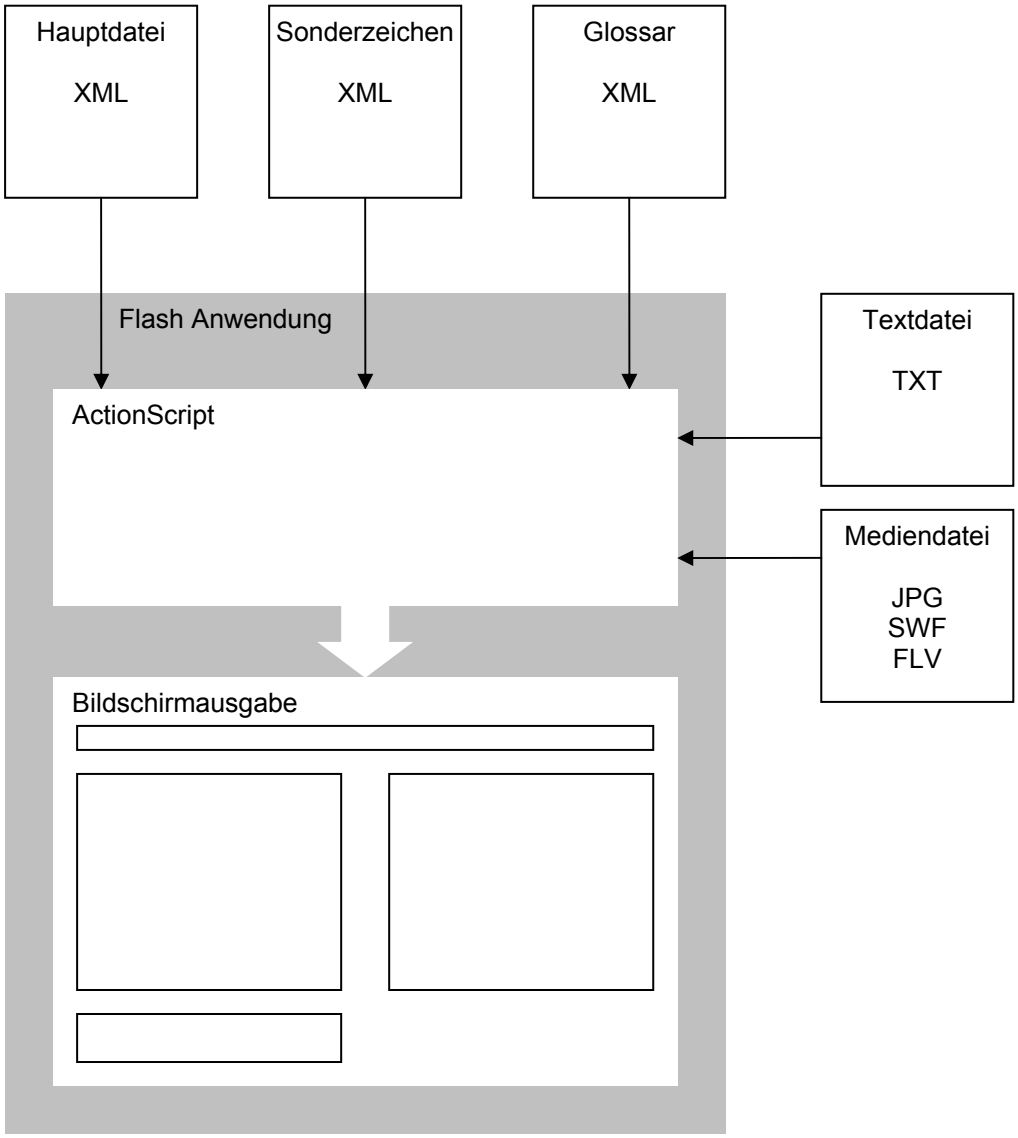


Abbildung 4.1.1: Interner Aufbau (simpel)

Darstellung

Dargestellt werden sollen Texte in Verbindung mit Mediendateien, wie Video-, Audio- und Bilddateien.

Bei den Mediendateien muss erst einmal grob unterschieden werden zwischen Bild und Audio/Videodateien. Ein Bild wird ganz normal als JPG-Datei in dem Format 500x480 Pixel gespeichert. Die Video- und Audiodateien allerdings sind zusammengesetzt aus einer SWF-Datei, also einem Flashfilm, und einer FLV-Datei bzw. MP3-Datei, die per SWF-Datei gestreamt werden.

Bei den Texten handelt es sich um Textdateien die in einem Ordner bereitliegen. Diese können mit einem ganz normalen Texteditor, wie zum Beispiel das Notepad, bearbeitet werden. Um Sonderzeichen wie die des griechischen Alphabets darzustellen, wird die Schreibweise von T_EX eingebunden.

Austausch von T_EX Elementen

Es stand die Frage im Raum, wie in Flash Symbole oder Sonderzeichen, wie sie das griechische Alphabet bietet, dargestellt werden.

Im wissenschaftlichen Bereich werden oft mathematische Formeln mit griechischen Buchstaben benötigt. Da diese Buchstaben auch oft in Texten vorkommen, zum Beispiel in Gleichungen, sollen sie natürlich auch dargestellt werden. Die wissenschaftlichen Texte werden zumeist in T_EX geschrieben. So war dann auch klar, dass die Zeichen des griechischen Alphabets in genau dieser Schreibweise in die Anwendung integriert werden sollen.

In T_EX werden spezielle Zeichen, zum Beispiel das griechische Alpha, durch eine Scriptschreibweise mit einem Backslash an erster Stelle dargestellt. Die T_EX-Schreibweise wird in den Textdateien angewandt und in der Anwendung durch das Sonderzeichen ersetzt.

In der Textdatei steht zum Beispiel:

```
Gleichung: n * \alpha = q * \beta
```

Aus dieser wird durch Umwandlung der Sonderzeichen:

```
Gleichung: n * α = q * β
```

4.2 Die Oberfläche

Die Grafische Benutzeroberfläche (GUI)

Die Grafische Benutzeroberfläche soll dem Benutzer nicht erst erklärt werden, sondern sie soll intuitiv benutzt werden können. Im Grunde soll sich der Benutzer vom ersten Aufruf der Anwendung, mit deren Look and Feel, wohlfühlen.

Die Navigation wird mit einem durchgängigen Menü mit Untermenüs im Drop-Down-Verfahren realisiert. Die Menüleiste ist horizontal angeordnet, so wie sie unter gängigen Windowsprogrammen auch zu finden ist.

Es gibt klar definierte Bereiche für Bilder und Videos, genau wie für die Texte. Diese befinden sich in der Mitte des Bildschirms, also im Hauptsichtbereich. Eine weitere Leiste befindet sich am unteren Rand. Dort werden die weiteren Themen angezeigt und daneben ein Bereich für weitere Navigationselemente, die schnell erreicht werden sollen.

Genauso wie es die Ergonomie am Arbeitsplatz gibt, gibt es auch die Software Ergonomie. Sie befasst sich mit der Benutzbarkeit von Computer-Programmen und deren Gebrauchstauglichkeit.

Eine verwirrende Navigation würde zum Beispiel Frustration und Stress hervorrufen. Um dies zu vermeiden wurde 1995 die EN ISO Norm 9241[16] eingeführt. In ihr wurden Kriterien zur Beurteilung von Software und Regeln zur besseren Gestaltung festgelegt. Eine davon ist zum Beispiel das Streben nach Konsistenz, das heißt, dass eine Benutzerschnittstelle möglichst so ausgelegt sein soll, dass der Benutzer möglichst wenig Überraschungen erlebt.

[16] de.wikipedia.org → Software-Ergonomie (Stand: 15.12.2004)

Die Benutzeroberfläche des E-Learnig-Tools ist im wesendlichen in neun Teilen Aufgeteilt.

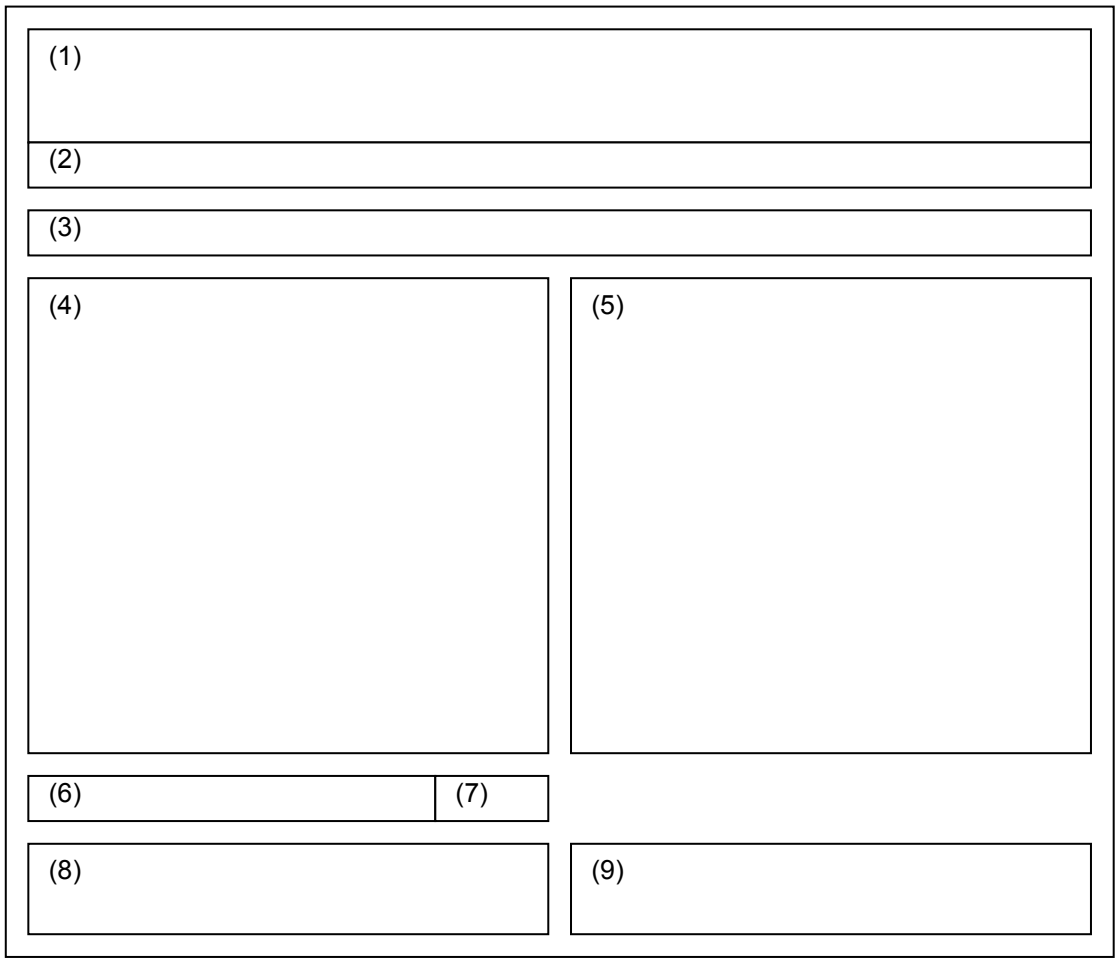


Abbildung 4.2.1: Skizze der Oberfläche

Im Folgenden wird sich auf Abbildung 4.2.1 bezogen.

(1) Logo / Firmengrafik

In (1) können Logo und Titel eingebunden werden. Dies ist eine Grafik, die nicht von einer Konfigurationsdatei gesteuert wird, da das System speziell für das HMI entwickelt wurde. Würde die Grafik mit der Konfigurationsdatei angepasst werden können, so könnte jeder einfach den Titel austauschen und die Anwendung nach seinen eigenen Bedürfnissen anpassen können. Dies ist aber nicht gewollt.

(2) Die Navigation

Eine klar definierte Navigation ist Voraussetzung. Wie in vorherigen Kapiteln schon angesprochen, wird es einen Einweisungs- und einen Theoriebereich geben. Dies sind auch die beiden Menüpunkte um die es sich in dem ganzen Projekt drehen wird.

Die Inhalte der beiden Punkte sind von außen, also außerhalb der Anwendung, definierbar. Die Unterpunkte werden per XML-Konfigurationsdatei aufgefüllt und während der Laufzeit beim Start der Anwendung, ausgewertet und gesetzt.

(3) Titelbereich

Hier wird der Titel zu dem jeweiligen Thema angezeigt. Jedes Thema, welches in der Navigation ausgewählt wird hat einen eigenen Titel.

(4) Medieninhalte

An dieser Stelle werden die Medien, d.h. Bilder, Videodateien oder auch Audiodateien, zu den jeweiligen Themen angezeigt. Die geplante Größe zur Einbindung ist 500x480 Pixel.

(5) Unterüberschrift + Textbereich

Die Texte zu den Themen werden in einem speziellen Textbereich dargestellt. Wird viel Text dargestellt, so erscheint ein Scrollbalken, ist es wenig Text, erscheint keiner. Vor dem Text wird noch eine Unterüberschrift zu dem Thema ausgegeben.

(6) Beschreibung des Medieninhalts

Zu jedem Medieninhalt gehört auch eine kurze Beschreibung, es ist eine Überschrift für das Bild oder die Audio-/Videoinhalte.

(7) Navigation für mehr als eine Mediendatei

In dem Bereich erscheint die Navigation falls es z.B. mehr als ein Bild zu diesem Thema gibt. Es kann mit der Navigation vor oder zurückgeblättert werden.

(8) Bereich für weiterführende Themen

So gut wie jedes Thema das angewählt werden kann, hat ein weiterführendes Thema. Hier werden die Überschriften dieser Themen angezeigt, die angewählt werden können.

(9) Platz für Buttons

An dieser Stelle ist Platz für weitere Buttons, wie z.B. ein Button zum Drucken oder Buttons für eine Historynavigation.

Das Glossar

Das Glossar soll in einem extra Fenster erscheinen, das sich durch einen einfach Klick auf das obere rechte Kreuz in dem kleinen Kästchen wieder schließen lassen soll. Während das Glossar erscheint, soll die darunter liegende Oberfläche nicht bedienbar sein.

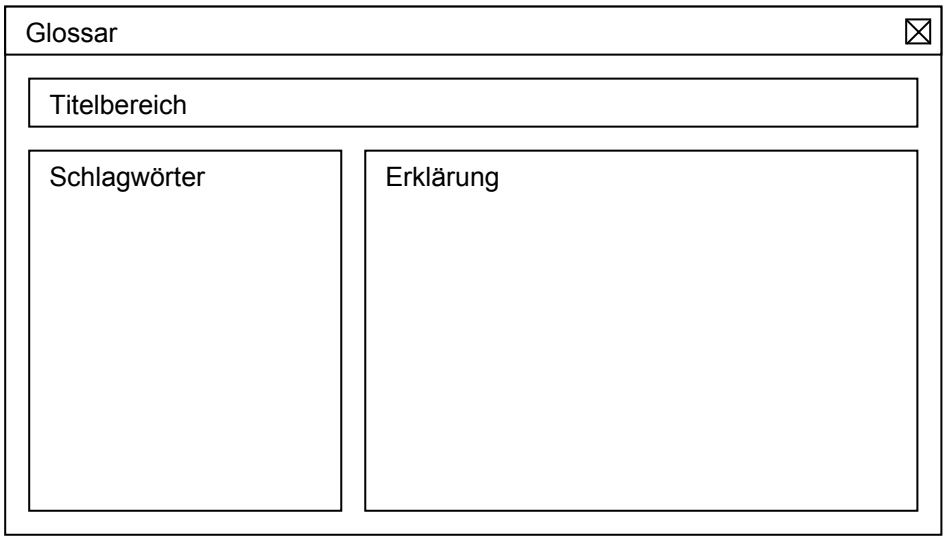


Abbildung 4.2.2: Skizze vom Glossarfenster

Das Fenster hat den Titel Glossar, der nicht verändert wird, im Titelbereich wird das jeweilig gewählte Schlagwort angezeigt, dessen Erläuterungen im Bereich angezeigt werden wo Erklärung steht (siehe Abbildung 4.2.2).

Auf der linken Seite werden die weiteren Schlagwörter des Glossars angezeigt, die mit einem Klick ausgewählt werden können.

5 Konzepte und Realisierung

Im Folgenden werden die Komponenten der Anwendung im Einzelnen erklärt. Als erstes wird der Aufbau der XML-Dateien ausführlich erläutert, da diese ein wesentlicher Bestandteil des ganzen Projektes sind.

Danach wird auf die Flashprogrammierung eingegangen um die interne Arbeitsweise der Anwendung zu erklären.

Zwischendurch werden immer wieder Teile der Oberfläche gezeigt um zu verdeutlichen wie sich externe Dateien auf sie auswirken.

Die Anwendung, d.h. die XML-Dateien, die ActionScript-Scripte und alle anderen Dateien müssen jetzt in einer Art CMS unter Flash zusammengefügt werden.

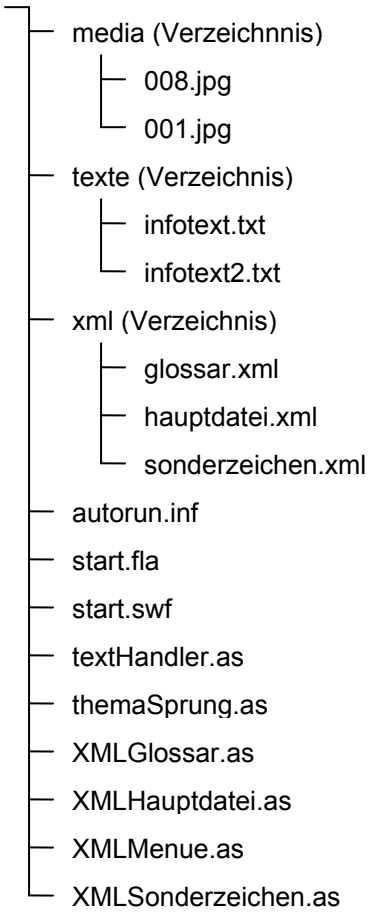


Abbildung 5.1: Verzeichnis während der Entwicklungszeit

In der Abbildung 5.1 ist die Verzeichnisstruktur während der Entwicklungsphase zu sehen. Die einzelnen Dateien werden alle durchgesprochen.

5.1 Aufbau der XML-Dateien

Das E-Learning-Tool basiert auf drei externen XML-Dateien, die Datei in der die Sonderzeichen für den dynamischen Austausch stehen, die XML-Datei in der die Glossareinträge verwaltet werden und der wichtigsten Datei, die Hauptdatei, in der die Themen eingetragen werden.

Die Hauptdatei

Die Datei "hauptdatei.xml" ist die zentrale Datei in der ganzen Anwendung. Existiert sie nicht, funktioniert die Anwendung nicht. Aus diesem Grunde hier noch einmal die Hauptaufgabe dieser Datei in einer schematischen Darstellung (Abbildung 5.1.1).

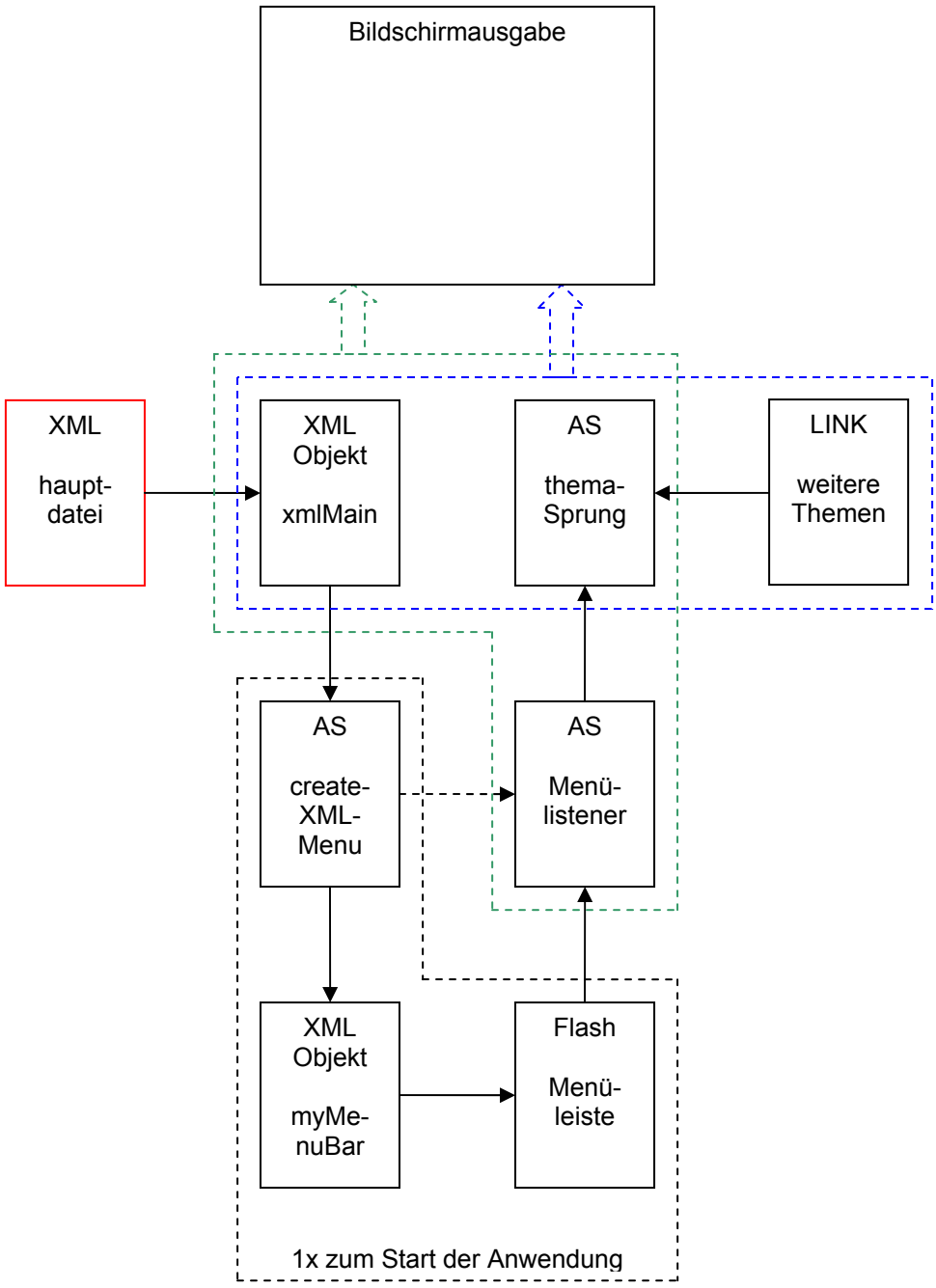


Abbildung 5.1.1: Aufgabe der Hauptdatei – Schematische Darstellung

Die Datei "hauptdatei.xml" wird nach dem Start der Anwendung einmalig per ActionScript (AS) in ein XML-Objekt, welches xmlMain benannt wurde, geladen.

Schwarz gestrichelter Bereich

Aus Teilen des Haupt-XML-Objekts wird per AS-Funktion dynamisch ein neues XML-Objekt mit dem Namen myMenuBar erzeugt. Dieses wird für die Flash-Komponente MenuBar gebraucht, da diese die Menüstruktur als XML-Objekt erwartet. Das erstellte XML-Objekt wird mit der Menubar-Komponente verlinkt und die Menüleiste wird angezeigt.

Grün gestrichelter Bereich

An der Menüleiste ist ein Menü-Listener gebunden. Dieser erzeugt bei Betätigen des Menüs, also das Anklicken eines Menüpunktes, ein Ereignis, welches abgefangen wird. Wurde ein neues Thema gewählt, so wird mittels der Funktion themaSprung und dem Haupt-XML-Objekt xmlMain die Bildschirmausgabe aktualisiert.

Blau gestrichelter Bereich

Genau wie durch die Wahl eines Menüpunktes, kann aber auch ein Link unter weitere Themen zu einem neuen Thema führen. Durch Klicken auf einen Link aus diesem Bereich wird auch die Funktion themaSprung aufgerufen, welche mit dem xmlMain-Objekt die Bildschirmausgabe aktualisiert.

Aber erst einmal zurück zur Datei "hauptdatei.xml". Zur Übersicht des Aufbaus der XML-Datei hauptdatei.xml ist in Listing 5.1.1 ein Datenknoten aus dieser abgedruckt.

```
<?xml version="1.0" encoding="UTF-8"?>
<main>
   <eintrag>
      <ueberschrift>Typische Anwendungen E2</ueberschrift>
      <unterueberschrift>Einige typische Anwendungen am E2</unterueberschrift>
      <media>
         <eintrag src="008.jpg">Smurf Eddison Verfahren</eintrag>
      </media>
      <text>infotext.txt</text>
      <weitereThemen>
         <eintrag>E2 Einführung</eintrag>
         <eintrag>E2 Messgeometrien</eintrag>
      </weitereThemen>
      <menuBar menu="Einweisung" submenu="E2" menuItem="Anwendungen"/>
      <index>
         <eintrag>E2 Messgeometrien</eintrag>
      </index>
   </eintrag>
   . . .
</main>
```

Listing 5.1.1: Aufbau hauptdatei.xml

Für einen besseren Überblick, welche Inhalte wo auf dem Bildschirm platziert werden, hier noch eine Abbildung (5.1.2).

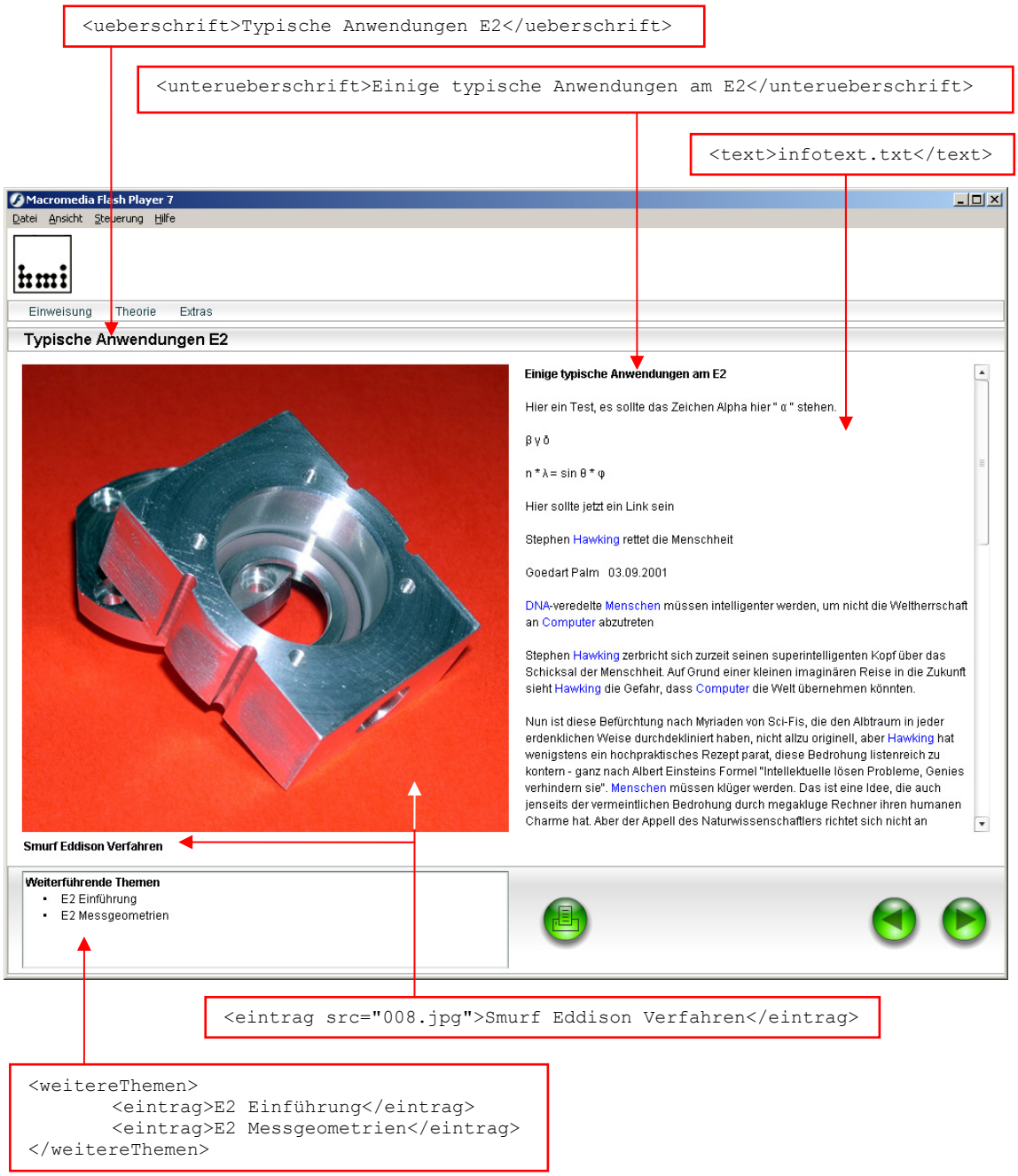


Abbildung 5.1.2: Platzierung der XML-Inhalte der hauptdatei.xml

In Listing 5.1.1 ist ein Datensatzeintrag der Haupt-XML-Datei zu sehen. Es wird nun Schritt für Schritt erklärt, wie sie aufgebaut ist, da der Administrator der Anwendung bzw. des E-Learning-Tools diese Datei am Häufigsten ändern wird.

Die Datei fängt mit einem üblichen XML-Kopf an, dieser wird allerdings nicht unbedingt für Flash benötigt. Aus Gründen der Wartung und Pflege ist er hier aber erhalten, da man hieraus erkennen kann, dass es sich um UTF-8 kodierte Zeichen handelt.

Der Hauptstrang fängt mit <main> an, danach kommen die eigentlichen Datenknoten, die mit <eintrag> eingeleitet werden.

```
<ueberschrift>Typische Anwendungen E2</ueberschrift>
```

Zwischen den Tags <ueberschrift> und </ueberschrift> wird die Überschrift platziert. Diese muss eindeutig sein und sollte auch nicht zu lang sein, d.h. sie darf kein zweites Mal benutzt werden. Wenn in der Anwendung von einem Thema zu einem anderen gesprungen wird so wird nach den Überschriften gesucht.

```
<unterueberschrift> Einige typische Anwendungen am
E2</unterueberschrift>
```

Diese Unterüberschrift wird vor dem eigentlichen Text etwas kleiner angezeigt, diese sollte genutzt werden, da sie etwas ausführlicher sein darf.

```
<media>
      <eintrag src="008.jpg">Smurf Eddison Verfahren</eintrag>
</media>
```

Unter dem Knoten <media> können die ganzen Mediendateien eingebunden werden. Jeder Eintrag wird von den Tags <eintrag src="{Dateiname}"> und </eintrag> umschlossen. In dem Attribut src="" muss zwischen den Anführungszeichen exakt der Dateiname des Bildes oder der SWF-Datei eingetragen werden, welches zu diesem Thema zur Verfügung stehen. Zwischen den </eintrag> Tags kommt eine Beschreibung der Mediendatei, damit Bild und der daneben stehende Text in Einklang gebracht werden können. Falls es mehrere Mediendateien zu einem Thema gibt, können mehrere Einträge, in der Art wie sie bei <weitereThemen> vorhanden sind, erstellt werden.

```
<text>infotext.txt</text>
```

Zwischen den <text> Tags kommt der Dateiname der Textdatei die diesem Thema angehört. Wie der Aufbau der Textdatei aussieht folgt weiter unten in diesem Kapitel.

```
<weitereThemen>
      <eintrag>E2 Einführung</eintrag>
      <eintrag>E2 Messgeometrien</eintrag>
</weitereThemen>
```

Dies ist ein Verweis auf ein anderes Thema. Wie schon bei dem Knoten <media> beschrieben, können hier auch mehrere Einträge erfolgen. Jeder Eintrag muss allerdings mit einem Eintrag aus <ueberschrift> genau übereinstimmen, da auch hier nach den Überschriften gesucht wird.

```
<menuBar menu="Einweisung" submenu="E2" menuItem="Anwendungen"/>
```

Einer der wichtigsten Einträge überhaupt in diesem Knoten. Es wird ein Menüeintrag in der Anwendung erzeugt, unter dem dieses Thema dann abrufbar ist. Es gibt drei Hauptmenüpunkte in der Anwendung, wobei hier nur zwei relevant sind, Einweisung und Theorie. Wenn in dem Eintrag menu="Einweisung" steht, kommt der Eintrag als Unterpunkt zu diesem Menüpunkt. Für menu kommen zwei Einstellungen in Frage, Einweisung oder Theorie.

Es kann ein Untermenüpunkt eingerichtet werden, mit `submenu="E2"` wird ein Untermenü mit den Namen E2 erstellt. Wird das Attribut `submenu` weggelassen oder ist es leer, so wird kein Untermenü eingerichtet.

Der eigentliche Menupunkt wird mit dem Attribut `menuItem` erstellt. Hier sollte man darauf achten, dass auch diese Menüpunkte eindeutig sein sollten, es sei denn, es werden die gleichen Menüpunkte in mehreren Untermenüs verwendet.

```
<index>
      <eintrag> E2 Messgeometrien</eintrag>
</index>
```

Dieser Eintrag ist schon für eine zukünftige Funktion in der Anwendung eingerichtet. Dieser Datensatzknoten ist für den Index im Programm zuständig. Hier werden Worte, die dem Thema entsprechen, eingetragen. Diese sollen dann in der Anwendung unter dem Punkt Index ausgewählt werden. Wird ein Indexpunkt gewählt, so wird zu dem Thema gesprungen. Es können mehrere Einträge gemacht werden. Damit der Administrator der Anwendung später nicht alle Einträge durchgehen muss, ist dieser Eintrag schon vorhanden. Nun braucht nur noch die Funktion implementiert werden und der Index ist schon vollständig.

Die Glossardatei

In dem Text eines Themas kommen immer wieder mal Begriffe vor, die nach einer Erklärung verlangen. Wie in einem HMTL-Dokument Begriffe mit einem Link versehen sind, so werden in der Anwendung die Begriffe, die in dem Glossar stehen, verlinkt.

Diese Begriffe sind in einer anderen Farbe als der Haupttext dargestellt. Jetzt kann der Anwender diesen auswählen und ein Glossarfenster öffnet sich. In diesem Fenster wird das ausgesuchte Schlagwort erklärt. Hier kann der Anwender auch alle anderen Schlagwörter sehen und auswählen um sich die Erklärungen durchzulesen.

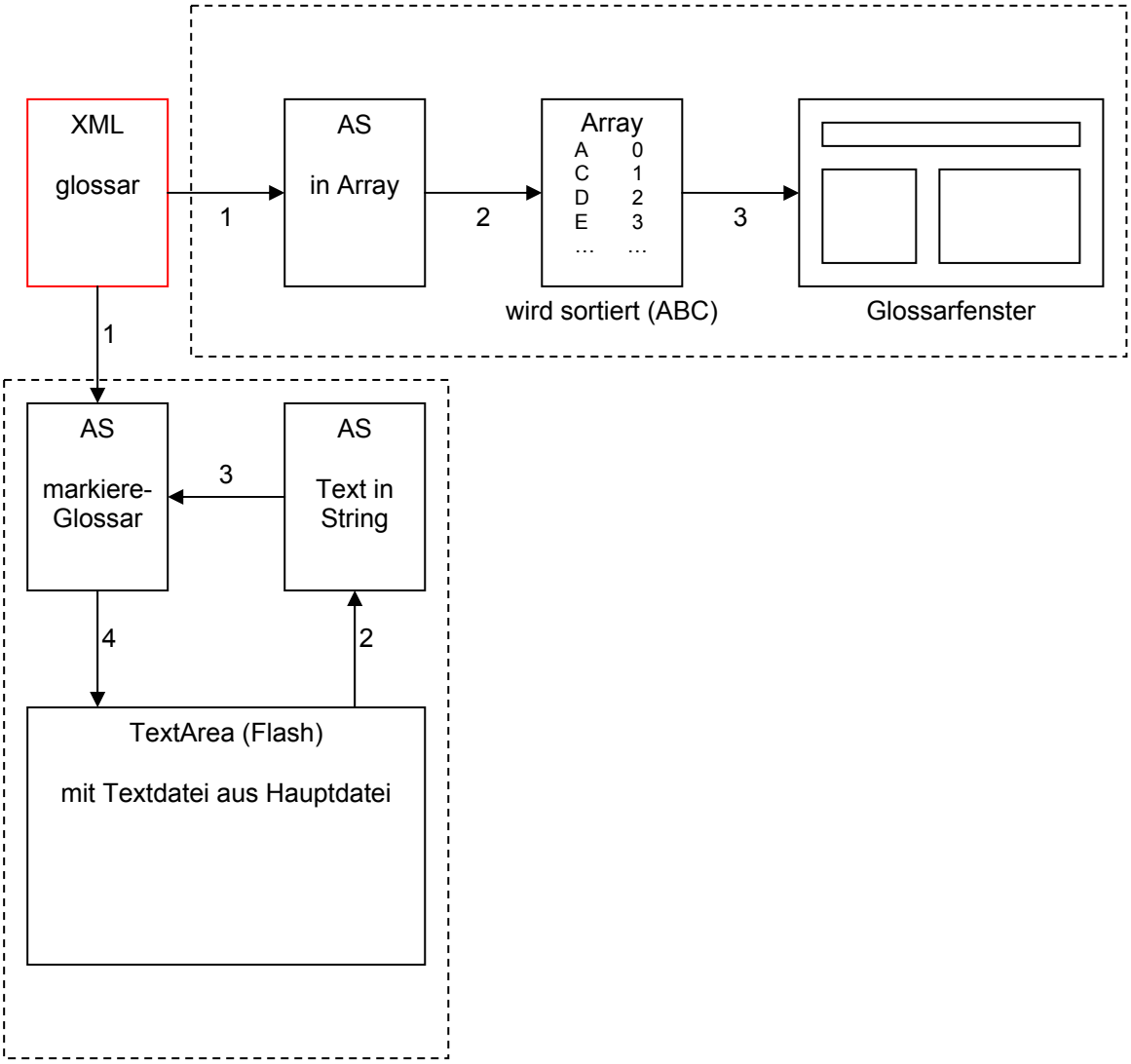


Abbildung 5.1.3: Aufgabe der Glossardatei – Schematische Darstellung

Die XML-Datei glossar.xml wird, wie die hauptdatei.xml, bei dem Start der Anwendung in ein XML-Objekt, welches xmlGlossar heißt, geladen. In dem rechten gestrichelten Bereich der Abbildung 5.1.3 ist zu sehen, dass die Glossareinträge über eine Funktion in ein Array gespeichert werden. Dies ist nötig, da das Glossarfenster eine Liste aller Glossareinträge enthält.

In dem unteren gestrichelten Bereich wird über eine Funktion markiereGlossar der Text aus der TextArea in einen String gespeichert und eine Funktion aufgerufen die die Schlagwörter aus dem Glossar im Text verlinkt (in Abbildung 5.1.2 blau im Text zu sehen) und den neuen Inhalt des Strings wieder zurück in die TextArea schreibt.

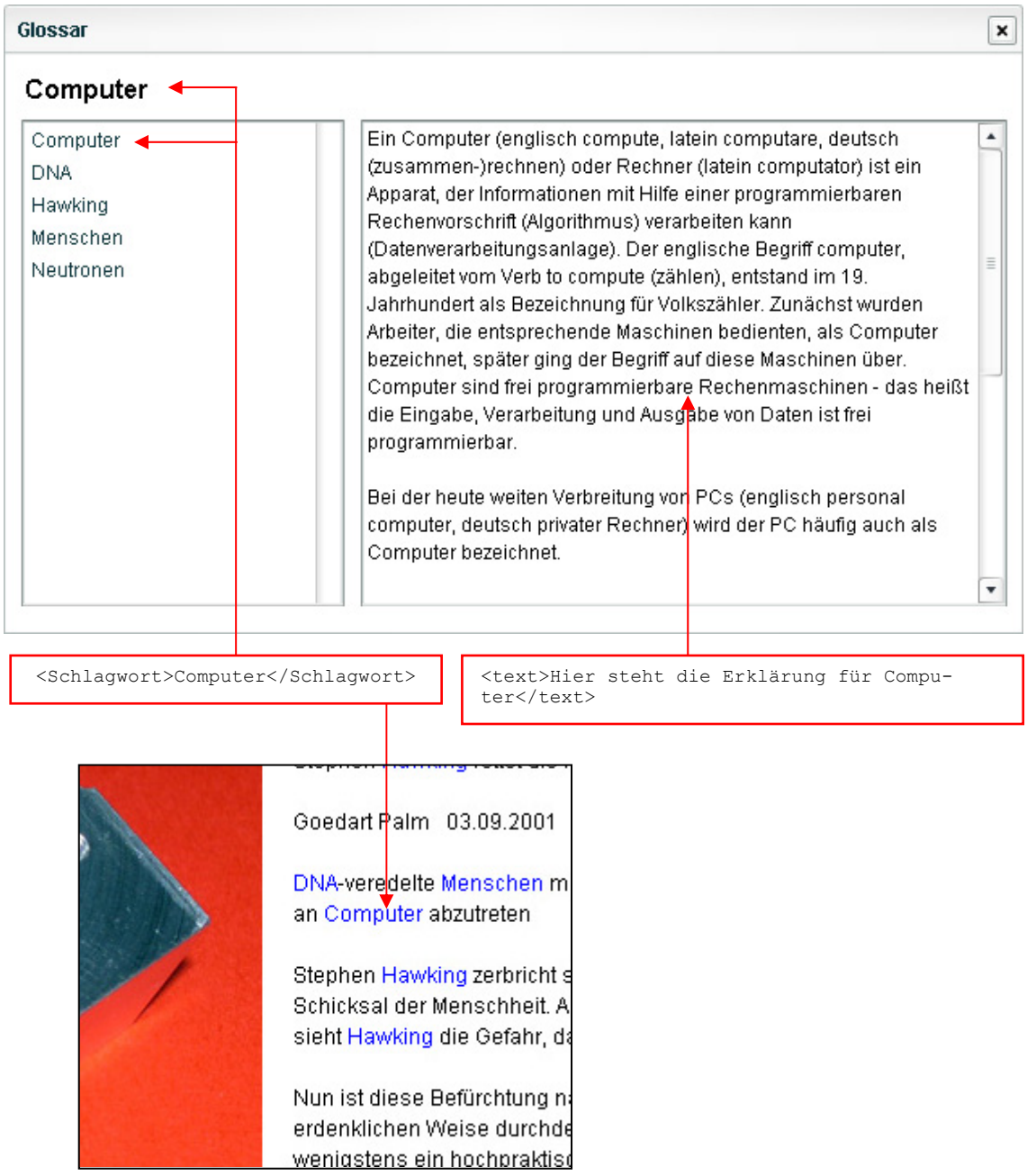

Glossar
Computer
Computer
DNA
Hawking
Menschen
Neutronen
Ein Computer (englisch compute, latein computare, deutsch (zusammen-)rechnen) oder Rechner (latein computator) ist ein Apparat, der Informationen mit Hilfe einer programmierbaren Rechenvorschrift (Algorithmus) verarbeiten kann (Datenverarbeitungsanlage). Der englische Begriff computer, abgeleitet vom Verb to compute (zählen), entstand im 19. Jahrhundert als Bezeichnung für Volkszähler. Zunächst wurden Arbeiter, die entsprechende Maschinen bedienten, als Computer bezeichnet, später ging der Begriff auf diese Maschinen über. Computer sind frei programmierbare Rechenmaschinen - das heißt die Eingabe, Verarbeitung und Ausgabe von Daten ist frei programmierbar.
Bei der heute weiten Verbreitung von PCs (englisch personal computer, deutsch privater Rechner) wird der PC häufig auch als Computer bezeichnet.
<Schlagwort>Computer</Schlagwort>
<text>Hier steht die Erklärung für Computer</text>
Goedart Palm 03.09.2001
DNA-veredelte Menschen m
an Computer abzutreten
Stephen Hawking zerbricht s
Schicksal der Menschheit. A
sieht Hawking die Gefahr, d
Nun ist diese Befürchtung n
erdenklichen Weise durchde
wenigstens ein hochpraktis

Abbildung 5.1.4: Das Glossarfenster und ein Ausschnitt aus der Anwendung

Abbildung 5.1.4 zeigt welche Informationen aus der XML-Datei glossar.xml im Glossarfenster und in der Anwendung dargestellt werden.

```
<?xml version="1.0" encoding="UTF-8"?>
<glossar>
    <eintrag>
       <Schlagwort>Computer</Schlagwort>
       <text>Hier steht die Erklärung für Computer</text>
    </eintrag>
    . . .
</glossar>
```

Listing 5.1.2: Aufbau glossar.xml

Genau wie bei der Haupt-XML-Datei wird auch hier (Listing 5.1.2), wie in jeder XML-Datei, mit einem XML-Kopf angefangen.

Es folgt der Hauptknoten, der die eigentlichen Datensätze enthält. Dieser wird mit `<glossar>` eingeleitet. Ein Eintrag entspricht dem, was zwischen `<eintrag>` und `</eintrag>` steht.

`<Schlagwort>Computer</Schlagwort>`

Die Einträge werden anhand des Schlagworts gesucht, welches zwischen dem gleichnamigen Tag steht. Dieses Schlagwort muss eindeutig sein und darf sich nicht als weiteres Schlagwort in dieser Datei befinden. Die Schlagwörter in der Datei werden für die Glossarliste, die in Abbildung 5.1.4 auf der linken Seite des Glossarfensters zu sehen ist, benutzt.

`<text>Hier steht die Erklärung für den Computer</text>`

Zwischen den Tags `<text>` wird die Erklärung des Schlagworts, wie es einem Glossar entspricht, erläutert. Auch dies ist anhand der Abbildung 5.1.4 zu sehen.

Die Sonderzeichen

Wie im vorherigen Kapitel beschrieben, werden die Sonderzeichen des griechischen Alphabets auch dargestellt. Sie werden in der TEX-Schreibweise in den Textdateien eingegeben und in der Anwendung dynamisch gegen das jeweilige Sonderzeichen ausgetauscht.

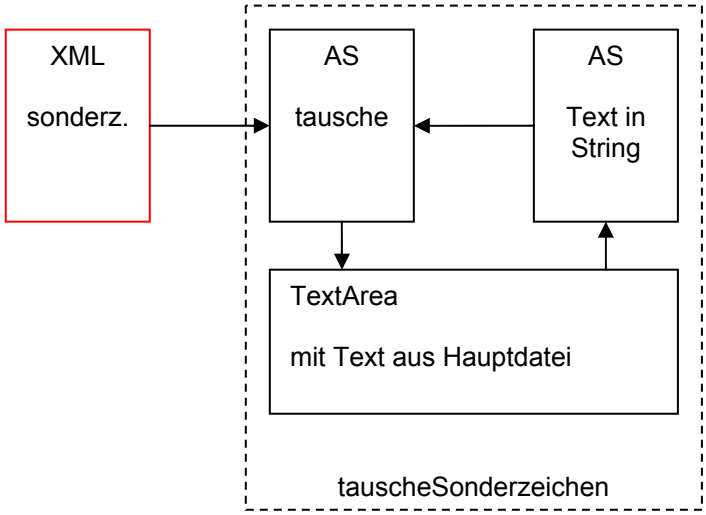


Abbildung 5.1.5: Aufgabe der Sonderzeichendatei – Schematische Darstellung

Die XML-Datei sonderzeichen.xml wird bei Start der Anwendung in ein XML-Objekt xmlSonderzeichen gespeichert. Die Funktion tauscheSonderzeichen holt sich aus der TextArea den Text und speichert ihn in ein String. Jetzt wird jedes TEX-Element mit dem Sonderzeichen aus dem XML-Objekt xmlSonderzeichen ausgetauscht und der neue Inhalt des String wird wieder in die TextArea kopiert.

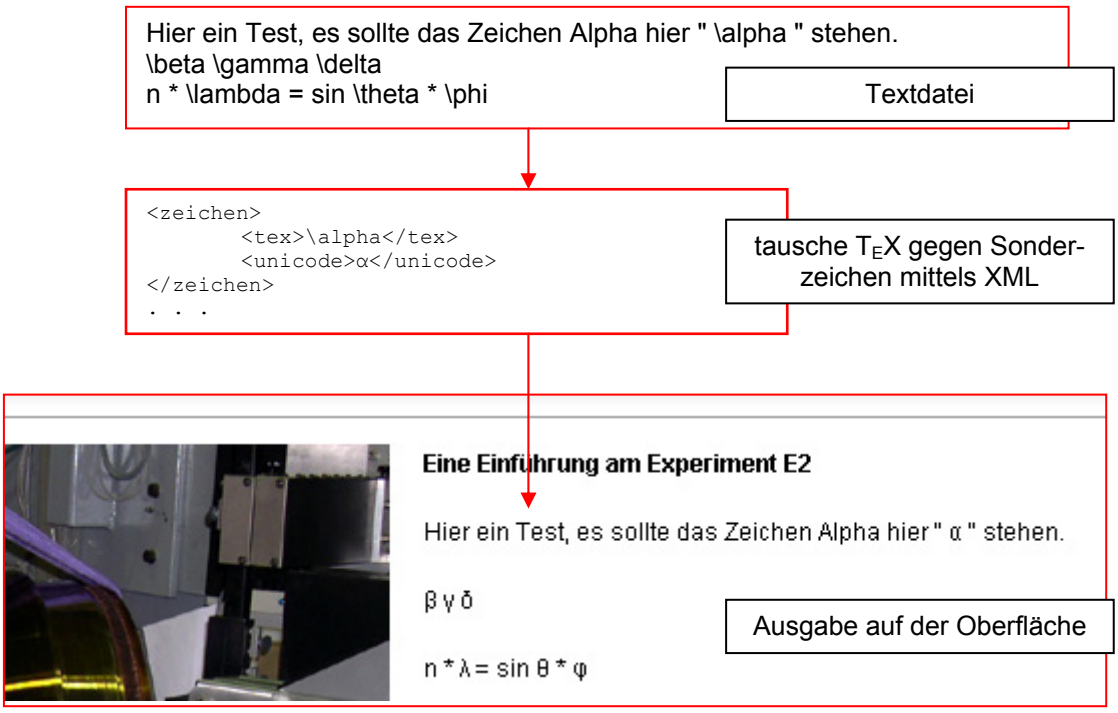


Abbildung 5.1.6: Austausch der Sonderzeichen

Der Austausch der T_EX-Elemente wird in Abbildung 5.1.6 noch einmal verdeutlicht.

```
<?xml version="1.0" encoding="UTF-8"?>
<sonderzeichen>
   <zeichen>
      <tex>\alpha</tex>
      <unicode>α</unicode>
   </zeichen>
   . . .
</sonderzeichen>
```

Listing 5.1.3: Aufbau sonderzeichen.xml

Die Datei "sonderzeichen.xml" ist ähnlich der vorherigen XML-Datei "glossar.xml" aufgebaut. Hauptknoten ist hier `<sonderzeichen>` und der Datensatzknoten `<zeichen>`.

`<tex>\alpha</tex>`

Das Element, welches durch ein Sonderzeichen, welches nicht so einfach über die Tastatur erreicht werden kann, wird hier eingetragen. Zu beachten ist auch hier, dass dieses Element nur einmal in der XML-Datei verwendet werden darf. Die anderen Elemente müssen unterschiedlich sein. Dem Element muss ein Backslash vorangestellt werden.

`<unicode>α</unicode>`

Dieses Tag beinhaltet das Sonderzeichen, welches mit dem oberen Element ausgetauscht wird. Um die Datei zu bearbeiten sei hier noch einmal angemerkt, dass ein Editor benutzt werden muss, der die Sonderzeichen darstellt.

Im Anhang (Kapitel 9) können die voreingestellten, d.h. die vorgegebene Sonderzeichen-Datei, eingesehen werden.

5.2 Die Strukturierung in Flash

Hier werden die Oberflächenelemente am Bild erläutert. Die jeweiligen Tabellen zeigen die Komponenten und deren Instanznamen, die in Flash verwendet wurden.

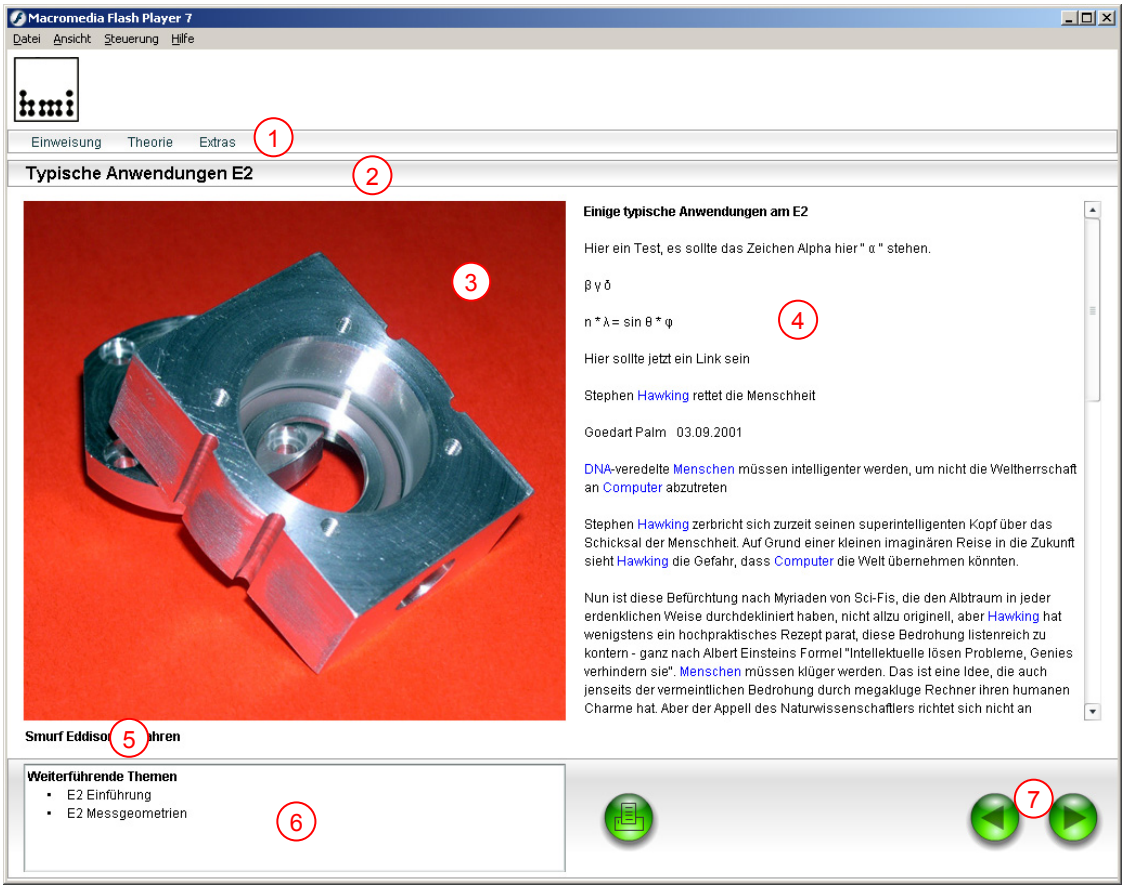

Macromedia Flash Player 7
Datei Ansicht Steuerung Hilfe
Einweisung Theorie Extras
1
Typische Anwendungen E2
2
3
Einige typische Anwendungen am E2
Hier ein Test, es sollte das Zeichen Alpha hier " α " stehen.
β γ δ
n * λ = sin θ * φ
4
Hier sollte jetzt ein Link sein
Stephen Hawking rettet die Menschheit
Goedart Palm 03.09.2001
DNA-veredelte Menschen müssen intelligenter werden, um nicht die Weltherrschaft an Computer abzutreten
Stephen Hawking zerbricht sich zurzeit seinen superintelligenten Kopf über das Schicksal der Menschheit. Auf Grund einer kleinen imaginären Reise in die Zukunft sieht Hawking die Gefahr, dass Computer die Welt übernehmen könnten.
Nun ist diese Befürchtung nach Myriaden von Sci-Fis, die den Albtraum in jeder erdenklichen Weise durchdekliniert haben, nicht allzu originell, aber Hawking hat wenigstens ein hochpraktisches Rezept parat, diese Bedrohung listenreich zu kontern - ganz nach Albert Einsteins Formel "Intellektuelle lösen Probleme, Genies verhindern sie". Menschen müssen klüger werden. Das ist eine Idee, die auch jenseits der vermeintlichen Bedrohung durch megakluge Rechner ihren humanen Charme hat. Aber der Appell des Naturwissenschaftlers richtet sich nicht an
Smurf Eddiso hren
5
Weiterführende Themen
E2 Einführung
E2 Messgeometrien
6
7

Abbildung 5.2.1: Verwendete Komponenten auf der Oberfläche

Nr	Name	Instanzname	Art	Verwendung
1	MenuBar	MenuLeiste	UI Komponente	Menüdarstellung
2	Textfeld	Ueberschrift	Dynamischer Text	Anzeige Überschrift
3	mc_platzhalter	platzhalter	Movieclip	Anzeige Mediadatei
4	TextArea	TextAusgabe	UI-Komponente	Für Textausgabe
5	Textfeld	bild_unterschrift	Dynamischer Text	Bildbeschreibung
6	TextArea	TextThemen	UI-Komponente	Weitere Themen
7	Schaltflächen	BT_...	Schaltfläche	verschieden

Abbildung 5.2.2: Liste der verwendeten Komponenten auf der Oberfläche (siehe Abbildung 5.2.1)

Und die gleiche Übersicht für das Glossarfenster.

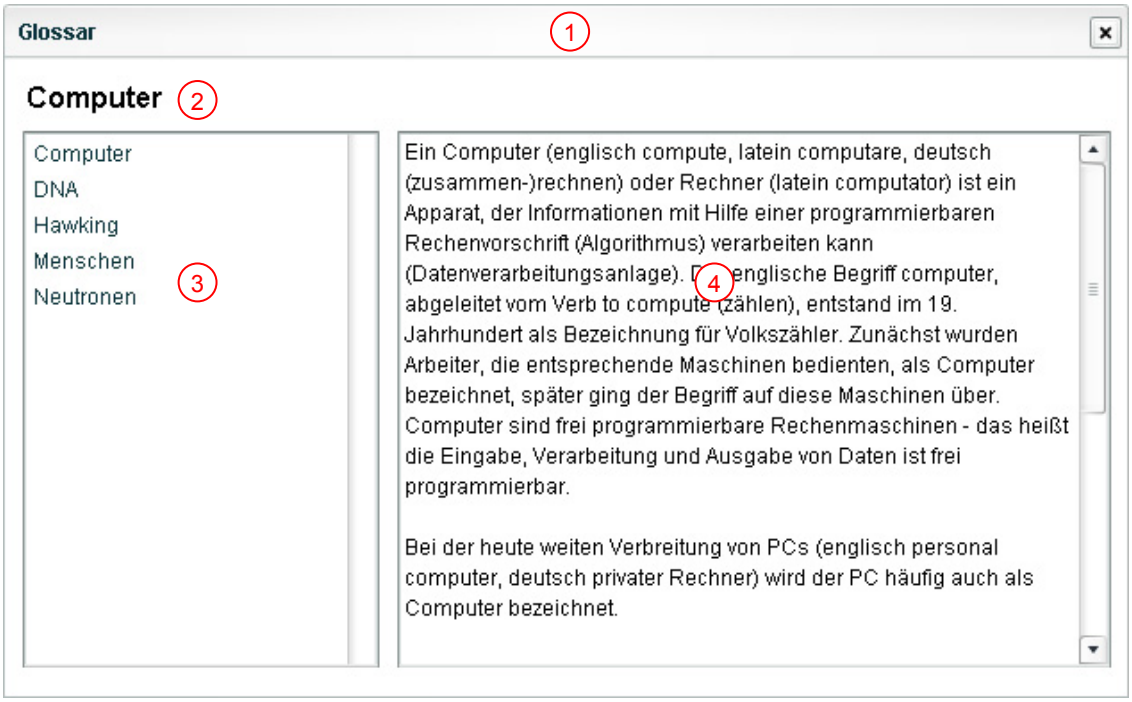

Glossar
1
Computer
2
Computer
DNA
Hawking
Menschen
Neutronen
3
Ein Computer (englisch compute, latein computare, deutsch (zusammen-)rechnen) oder Rechner (latein computator) ist ein Apparat, der Informationen mit Hilfe einer programmierbaren Rechenvorschrift (Algorithmus) verarbeiten kann (Datenverarbeitungsanlage). Der englische Begriff computer, abgeleitet vom Verb to compute (zählen), entstand im 19. Jahrhundert als Bezeichnung für Volkszähler. Zunächst wurden Arbeiter, die entsprechende Maschinen bedienten, als Computer bezeichnet, später ging der Begriff auf diese Maschinen über. Computer sind frei programmierbare Rechenmaschinen - das heißt die Eingabe, Verarbeitung und Ausgabe von Daten ist frei programmierbar.
4
Bei der heute weiten Verbreitung von PCs (englisch personal computer, deutsch privater Rechner) wird der PC häufig auch als Computer bezeichnet.

Abbildung 5.2.3: Verwendete Komponenten im Glossarfenster

Nr	Name	Instanzname	Art	Verwendung
1	Window	Win (Objekt)	UI Komponente	Glossarfenster selber
2	Textfeld	glossarTitel	Dynamischer Text	Anzeige Überschrift
3	List	glossarList	UI-Komponente	Anzeige Glossareintäge (Schlagwörter)
4	TextArea	Ausgabe	UI-Komponente	Für Textausgabe

Abbildung 5.2.4: Liste der verwendeten Komponenten im Glossarfenster (Abbildung 5.2.3)

5.3 Die Programmierung in ActionScript

Unterschied AS 1 und AS 2

Im Gegensatz zur ersten Version von ActionScript hat die Version 2.0 eine Menge an Fähigkeiten hinzugewonnen. Die OOP-Fähigkeiten wurden in der neuen Version ausgebaut. Jetzt kann man in ActionScript Klassen erstellen und diese wie in Java gewohnt einbinden. Des Weiteren können jetzt ActionScript-Dateien, die Code-Fragmente enthalten, erstellt werden und importiert werden.

Macromedia hat mit ActionScript 2 die strikte Typisierung eingeführt, diese ist aber nur optional. In Flash MX 2004 (Professional) kann der Entwickler sich entscheiden, ob er im Stil von ActioScript 1 oder 2 programmieren möchte. Selbst ein Gemisch aus beiden ist erlaubt.

Die Entwickler werden aufgefordert, den Datentyp bei der Definition einer Variablen anzugeben. Dazu einige Beispiele.

Um eine variable Zahl zu definieren war es in ActionScript 1 möglich, folgenden Code anzugeben:

```
var zahl;
```

In ActionScript 2 muss (soll) jetzt der Typ mit angegeben werden. Das sieht so aus:

```
var zahl:Number;
```

Für Strings sieht es genau so aus. Statt einfach die Variable zu definieren ohne Typangabe wird ein `:String` angehangen. Das gilt auch für alle anderen Datentypen.

Funktionen werden in ActionScript 2 auch anders definiert. In ActionSript 1 sah dies so aus:

```
function tuwas() {. . .}
```

Diese Funktion macht irgendetwas, es kann aber nicht abgelesen werden, ob sie etwas zurückgibt. Der Code der Funktion muss also durchgegangen werden. In ActionScript 2 sieht das anders aus:

```
function tuwas(Void):Void {. . .}
```

Zwischen den Klammern steht `Void`. Daran kann erkannt werden, dass die Funktion keinen Übergabewert braucht. Nach den Klammern folgt ein Doppelpunkt mit einem `Void`. Dieses ist das Erkennungszeichen, dass die Funktion keinen Wert zurückgibt. Eine Funktion, die zum Beispiel eine Zahl als Argument übergeben bekommt und einen String als Rückgabewert liefert, sieht folgendermaßen aus:

```
function tuwas(zahl:Number):String {. . .}
```

Aus der strikten Typisierung folgt also, dass eine bessere Übersichtlichkeit gegeben ist und es können Fehler durch falsche Zuweisungen vermieden werden.

Dies fordert aber den Entwicklern strenge Disziplin ab. Wenn also einfach alter ActionScript 1 Code eingebunden wird, was sich zur Zeit an manchen Stellen einfach nicht verhindern lässt, da ActionScript 2 noch zu neu ist und Flash es in dieser Version erlaubt, dann greift die strikte Typisierung nicht.

XML-Dateien in Flash einbinden

Nachdem der Aufbau der XML-Dateien in Kapitel 5.1 erklärt wurde, müssen diese natürlich auch in Flash eingebunden werden. Wie geht das?

Wenn es elegant gemacht werden soll, dann natürlich mit einer Klasse. Das Problem daran ist, dass das XML-Objekt dann nicht sichtbar ist und nur der Klasse zur Verfügung steht. Nach einigen Stunden Recherche in Büchern (siehe Literaturliste) und im Internet im Flashforum stand immer noch keine Lösung für das Problem zur Verfügung. Selbst bis zum Druck dieser Arbeit ist immer noch keine Lösung gefunden worden. Einen ersten Ansatz bietet aber das Listing im Anhang (Kapitel 9).

Also wurde das Projekt komplett ohne Klassen, also nur mit Funktionen und externen ActionScript-Dateien, umgesetzt. Dadurch existiert wenigstens ein modularer Aufbau, denn ein Mix aus Klassen und klassischer ActionsScript-Programmierung ist verwirrender als ohne Klassen. Entweder ganz oder gar nicht.

Natürlich werden aber die von Flash zur Verfügung gestellten Klassen zur Manipulation von Strings, Arrays und die Komponenten verwendet.

Folgende XML-Dateien müssen in die dazugehörigen XML-Objekte geladen werden, danach können diese benutzt werden.

Datei	XML-Objekt
hauptdatei.xml	xmlMain
sonderzeichen.xml	xmlSonderzeichen
glossar.xml	xmlGlossar

Abbildung 5.3.1: XML-Datei → XML-Objekt

```
var xmlMain:XML = new XML();
xmlMain.onLoad = xmlMainGeladen;
xmlMain.ignoreWhite = true;
xmlMain.contentType = "text/xml";
xmlMain.load(src_xml+"hauptdatei.xml");

function xmlMainGeladen(erfolgreich):Void {
 if(erfolgreich) {
        createXMLMenu();
 } else {
        trace ("Hauptdatei -> Fehler");
 }
}
```

Listing 5.3.1: Laden der hauptdatei.xml in das XML-Objekt xmlMain

Gegenüber Kapitel 3.5, wo gezeigt wurde wie eine XML-Datei in ein dazugehöriges XML-Objekt gespeichert wird, hat sich hier folgendes geändert.

```
xmlMain.contentType = "text/xml";
```

Mit dieser Zeile wird Flash veranlasst, dass der Typ der Daten im Objekt, XML-Daten sind. Ist die Datei erfolgreich im XML-Objekt `xmlMain` gespeichert, so wird die Funktion `createXMLMenu()` aufgerufen. Dazu später mehr.

Genau wie im Listing 5.3.1 werden auch die anderen beiden XML-Dateien (siehe Abbildung 5.3.1) behandelt.

Darstellung von Text in Flash

In Flash gibt es mehrere Möglichkeiten Text darzustellen. Es gibt das Textwerkzeug aus der Werkzeugleiste, mit dem Textfelder erstellt werden und TextAreas, die als Flash-Komponente zur Verfügung stehen.

Mit dem Textwerkzeug lassen sich drei unterschiedliche Arten von Text erstellen (siehe Abbildung 5.3.2 und 5.3.3).

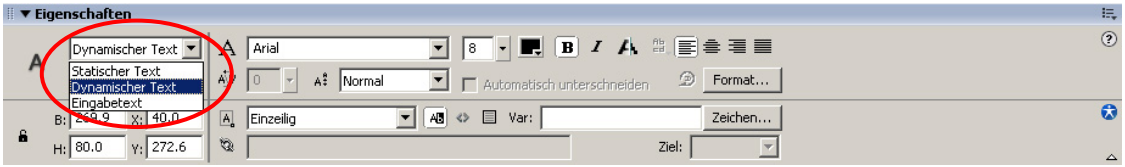


Abbildung 5.3.2: Eigenschaften eines Textfeldes

Arten von Text mit dem Textwerkzeug	
Statischer Text	Text wird beim Erstellen des Films festgelegt und lässt sich während der Film läuft nicht verändern.
Dynamischer Text	Der Text kann bei der Erstellung des Films vorgegeben werden und ist während des Ablaufs des Films über ActionScript veränderbar.
Eingabetext	Ein Textfeld in dem der Benutzer eine Eingabe machen kann. Es lässt sich per ActionScript abfragen.

Abbildung 5.3.3: Arten vom Einsatzes des Textwerkzeuges

Das Textwerkzeug eignet sich am besten für Überschriften oder andere einzeilige Texte, wie zum Beispiel eine Beschreibung für ein Bild etc. Möchte der Entwickler mehr als nur eine Zeile ausgeben so muss er die Größe des Textfeldes vorher definieren und wenn er den Text durch einen längeren Text austauscht, die Komponente UIScrollBar [17]mit an das Textfeld heften.

Der Autor hat die Erfahrung gemacht, wenn ein vorher definiertes Textfeld mit UIScrollBar nicht mit Text aufgefüllt wurde und während der Laufzeit ein langer Text in das Textfeld geladen wurde, dass der Scrollbalken nicht erscheint. Nach einigen Versuchen und Diskussionen in einem Internet Forum, das Flashforum[18], wurde auf die neu in Macromedia Flash MX 2004 (Professional) dazugekommene UI Komponente TextArea umdisponiert.

[17] UI für User Interface
[18] http://www.flashforum.de (Stand: 17.12.2004)

Die TextArea ist leicht handhabbar. Der Entwickler braucht nur die UI Komponente auf die Bühne ziehen und kann sie zurechtschieben. In den Eigenschaften kann jetzt das Allernötigste eingestellt werden (Abbildung 5.3.4).

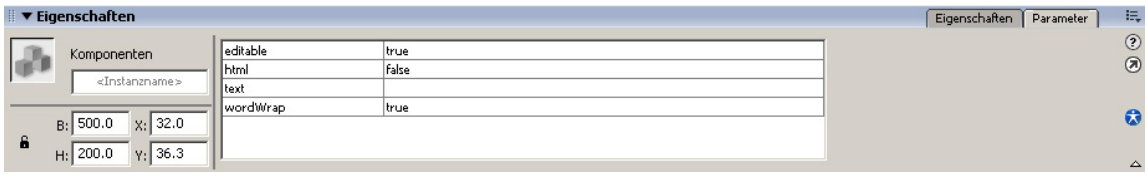


Abbildung 5.3.4: Eigenschaften einer TextArea

Die TextArea setzt die Scrollbar automatisch, es kommt nur darauf an wie viel Text in die TextArea platziert wird.

```
TextAusgabe.text = "Das ist der Text.";
```

Mit dieser Zeile wird eine TextArea mit dem Instanznamen `TextAusgabe` mit dem Text "Das ist der Text." gefüllt. Genauso gut kann dort eine Variable angegeben werden, in der noch viel mehr Text steht.

Eine wichtige Funktion für das Projekt ist, dass der Text als HTML-Text verarbeitet wird. Dazu werden in ActionScript folgende Befehlszeilen eingegeben:

```
// Textausgaben auf HTML setzen
TextAusgabe.html = true;
TextThemen.html = true;
```

Listing 5.3.2: TextAreas auf HTML-Text setzen

So versteht eine TextArea auch HTML-Tags.

```
TextAusgabe.text = "<b>Das ist der Text.</b>";
```

Der in der TextArea `TextAusgabe` dargestellte Text wird fett dargestellt. Damit ist es auch möglich, Links per `<a href="">` zu erzeugen.

Das Problem ist, wenn im Text eine Verlinkung mit `<a href="">` erstellt wird, wie kann ich eine bestimmte Funktion anspringen? Die Lösung wurde vom Autor nach einiger Zeit gefunden, sie heißt `asfunction`.

```
<a href='asfunction:eineFunktion,argument'> Link </a>
```

Wenn dieses in der TextArea eingebunden wird, so wird das Wort Link dargestellt und es kann angeklickt werden. Daraufhin wird die ActionScript-Funktion `eineFunktion` mit dem Argument `argument` aufgerufen.

Genau das wird für die Glossareinträge und den weiteren Themen gebraucht. Aber die Einträge, die ersetzt werden sollen, müssen irgendwie ausgetauscht werden.

Textdateien laden

Im E-Learning-Tool werden die Texte für die Darstellung nachgeladen. Dieses wird mit der LoadVars-Klasse durchgeführt. Texte werden in einer Variablen. die mit in der Datei steht, gespeichert. Die Dateien müssen wieder im UTF-8-Format gespeichert werden. Am Anfang des Textes muss der Variablenname gefolgt von einem Gleichheitszeichen stehen. In dem Fall für Listing 5.3.3 heißt die Variable var1. Der Inhalt der Datei muss zum Beispiel wie folgt aussehen.

```
var1=Hier steht der Text…
```

Um diesen Text in Flash einzubinden wird folgender ActionScript Code (Listing 5.3.3) benötigt.

```
var loadVarsText = new LoadVars();
loadVarsText.load("textfile.txt");

loadVarsText.onLoad = function(success) {
    if (success) {
       TextAusgabe.text = this.var1;
    } else {
         trace("not loaded");
    }
}
```

Listing 5.3.3: Laden einer Textdatei "textfile.txt"

Es wird ein neues LoadVars-Objekt `loadVarsText` erzeugt. Anschließend können mit diesem Objekt die Methoden der Klasse benutzt werden. Geladen wird der Text mit der `load(…)`-Methode welche per `onload`-Funktion überwacht wird. Bei erfolgreichem laden wird der Text in der Variable `var1` gespeichert. Diese kann zum Beispiel einem Textfeld oder besser noch, einer TextArea mit `this.var1` übergeben werden.

Funktion suchen und ersetzen

Zum Ersetzen der Glossareinträge und der T_EX Codes wird eine Funktion benötigt, die in einer TextArea Text ersetzen kann. Da aber der Befehlsumfang von ActionScript 2 keinen Befehl zum Suchen und Ersetzen in Strings hat, muss die Funktion selber programmiert werden.

```
String.prototype.replace=function(suchstr:String,ersetzstr:String):String {
     return this.split(suchstr).join(ersetzstr);
}
```

Listing 5.3.4: Funktion zum Suchen und Ersetzen

In Listing 5.3.4 ist eine Funktion zum Suchen und Ersetzen zu sehen. Sie benötigt ein String zum Suchen und einen, der für diesen ersetzt wird. Die Methode `split()` gehört der Klasse String an. Mit ihr kann ein String an den Stellen geteilt werden, an denen der übergebene String `suchstr` steht. Sie liefert die einzelnen Teilstrings so als ein Array zurück.

Die darauf folgende Methode `join()` aus der Array Klasse, wandelt das Array wieder zurück in ein String. Zwischen den einzelnen Array-Elementen wird der `ersetzstr` gesetzt, woraufhin in dem übergebenen String, an jeder Stelle wo der Suchstring stand, jetzt der Ersetzstring steht.

Mit folgendem Code in Listing 5.3.5 kann die Funktion nun angewandt werden. Mtext ist der String, in dem der Ausgangstext steht.

```
Mtext = _root.Text.TextAusgabe.text;
Mtext = Mtext.replace("\\alpha","\u03B1");
_root.Text.TextAusgabe.text = Mtext;
```

Listing 5.3.5: Funktion aus Listing 5.3.4 anwenden

In einem Text, in dem jetzt ein "\alpha" steht, wird jetzt das Sonderzeichen aus dem Unicode-Zeichensatz, das α. Da der Backslash in dem Sourcecode einem Sonderzeichen vorangestellt wird, muss dort natürlich ein doppeltes Backslash stehen.

Wenn jetzt der Funktion nicht direkt ein String, in diesem Falle "\\alpha" und "\u03B1", übergeben wird, sondern ein String, der diese als Inhalt hat, so steht statt dem Sonderzeichen "α" der Inhalt des Strings, nämlich "\u03B1" an der zu ersetzenden Stelle.

Aus irgendeinem Grund kann man als String in dieser Funktion keine Unicodes wie /uxxx angeben, sondern muss das schon dekodierte Zeichen in dem String übergeben. Wenn man allerdings den Unicode "/uxxx" direkt beim Funktionsaufruf angibt, funktioniert dieses hervorragend.

Dieses Phänomen kann jemanden schier zur Verzweiflung bringen, weil es einem das schön durchdachte Konzept zerstört. Dadurch muss jetzt ein Editor verwendet werden, der erstens Sonderzeichen darstellen kann und zweitens, die Datei als UTF-8 abspeichern kann.

```
function markiereGlossar(Void):Void {

   // Referenz auf Knoten
   var Knoten:XMLNode = _root.xmlGlossar.firstChild;

   Mtext = _root.TextAusgabe.text; // Textfeld in String schreiben

   for (var i:Number = 0; i < Knoten.childNodes.length; i++) {
      Mtext = Mtext.
      replace(Knoten.childNodes[i].childNodes[0].firstChild.nodeValue,
      "<a href='asfunction:_root.glossarLink," + i + "'><font color='" +
      _root.GlossarEintragFarbe + "'>" +
      Knoten.childNodes[i].childNodes[0].firstChild.nodeValue +
      "</font></a>");
   }
   _root.TextAusgabe.text = Mtext; // String in Textfeld zurückschreiben
}
```

Listing 5.3.6: Funktion zur Verlinkung der Glossareinträge

Der ActionScript-Code zum Austausch der Glossareinträge ist in Listing 5.3.6 zu sehen.

Als allererstes wird eine Referenz auf das XML-Objekt `xmlGlossar` erstellt und der Text ein String geschrieben.

In der for-Schleife wird jedes Schlagwort im XML-Knoten durchlaufen und jedes Mal das Schlagwort in dem String ersetzt. Ersetzt wird es durch ein Link auf das Schlagwort, welcher die Funktion `glossarLink` aufruft. Übergeben wird ein Argument, in diesem steht die Stelle an welcher das Schlagwort in dem XML-Objekt xmlGlossar steht. Wenn der Link in der so präparierten TextArea aufgerufen wird, kann jetzt schnell zu dem gewünschten Schlagwort gesprungen werden und die dazu passende Erklärung ausgegeben werden.

Dazu wird mittels der Funktion `glossarLink` und der UI Komponente Window das Glossarfenster geöffnet und der passende Eintrag zum Schlagwort ausgegeben.

```
_root.GlossarText =

Knoten.childNodes[arg].childNodes[1].firstChild.nodeValue;
```

In dem Argument welches übergeben wurde, in der Funktion `glossarLink` ist es `arg`, kann die Stelle im XML-Knoten genau ermittelt werden und der Text in die TextArea des Glossarfensters wird ausgegeben.

So wie die Glossareinträge ausgetauscht werden, müssen natürlich auch die Sonderzeichen ausgetauscht werden. Mit der Funktion `tauscheSonderzeichen` wird dies im gleichen Maße umgesetzt wie bei dem Tausch der Glossareinträge.

Menüstruktur mit der UI Komponente MenuBar

Die Menüstruktur mit der UI Komponente MenuBar war ein weiteres Ziel die Integration der Flash-Komponenten zu nutzen. Dies erwies sich als wahrer Knackpunkt der ganzen Anwendung. Aber zuerst einmal sollte der Umgang mit der UI Komponente MenuBar erklärt werden.

Abbildung 5.3.5: Die Menüleiste

Die MenuBar-Komponente braucht zur Generierung der Menüpunkte ein XML-Objekt, in dem die Einträge für das Menü stehen. Es werden folgende Punkte benötigt.

Attribut	Erklärung
name	Hier wird angegeben wie der Menüpunkt heißen soll.
instanceName	Für die Ereignisbehandlung in ActionScript benötigter Referenzname
listener	Mit diesem Namen wird die Ereignisbehandlungsrouzine später erstellt
label	Name des Menüpunktes, also der Eintrag unter dem der Menüpunkt erscheint.

Abbildung 5.3.6: Wichtige Einträge zur Erstellung der Menüpunkte

Ein Menü mit Flash erstellen

Es wird im Folgenden beschrieben, wie eine Menüleiste mit einem Menüpunkt "Datei" und dem Eintrag "Öffnen" erstellt wird.

Voraussetzung zur Erstellung eines Menüs mit der MenuBar-Komponente ist natürlich das Autorensystem Macromedia Flash MX 2004 Professional.

Eine XML-Datei mit dem Namen "menubar.xml" muss erstellt werden und im UTF-8-Format gespeichert werden. In dieser werden die Bestandteile des Menüs eingetragen (Listing 5.3.7).

```
<?xml version="1.0" encoding="UTF-16"?>
<menuBar>
    <menu name="Datei" instanceName="datei" listener="dateiListener">
       <menuItem label="Öffnen" instanceName="datei_oeffnen"/>
    </menu>
</menuBar>
```

Listing 5.3.7: Ein kleines Menü – Datei "menuBar.xml"

1. Öffnen Sie ein neues Flash-Dokument.

2. Ziehen Sie aus den UI Components die MenuBar auf die Bühne.
3. Geben Sie der MenuBar über die Eigenschaften einen Instanznamen. In diesem Fall wird der Name myMenu genommen.
4. Klicken Sie auf das erste Bild in der Zeitleiste und öffnen danach den ActionScript-Editor (Aktionen).
5. Geben Sie folgenden Code aus Listing 5.3.8 ein.

```
1   var menuXML = new XML();
2   menuXML.ignoreWhite = true;
3   menuXML.contentType = "text/xml";
4   menuXML.load("menubar.xml");
5
6   menuXML.onLoad = function(success){
7      if(success){
8         // Länge des XML-Knotens (wieviele Menüeinträge?)
9         var laenge:Number = menuXML.firstChild.childNodes.length;
10        for (var i:Number = 0; i < laenge; i++) {
11           myMenu.addMenu(menuXML.firstChild.firstChild.attributes.name,
12           menuXML.firstChild.firstChild);
13        }
14     } else {
15        trace("Fehler beim Laden der XML-Datei");
16     }
17  }
```

Listing 5.3.8: XML-Datei mit menuBar verbinden

Der ActionScript-Code aus Listing 5.3.8 sorgt dafür, dass die XML-Datei "menubar.xml" geladen wird und die Menüleiste die Menüpunkte zugewiesen bekommt.

In den Zeilen 1 bis 4 wird das XML-Objekt menuXML erstellt und die Datei menubar.xml in diesem, wie in Kapitel 3.5 beschrieben, gespeichert. Neu sind die Zeilen 8 bis 13.

```
var laenge:Number = menuXML.firstChild.childNodes.length
```

Eine Variable `laenge` wird definiert und ihr wird die Anzahl der `<menu>`-Einträge, also der Menüpunkte - in diesem Falle ist es nur einer - übergeben. Daraufhin werden die einzelnen Einträge mit der For-Schleife durchlaufen und für jeden Eintrag das Menü hinzugefügt. Dies geschieht mit folgendem Befehl.

```
myMenu.addMenu(menuXML.firstChild.firstChild.attributes.name,
menuXML.firstChild.firstChild);
```

Aus der MenuBar-Klasse, welche Flash zur Verfügung stellt, wird die Methode `addMenu` benutzt. Das erste Argument, welches sie bekommt, ist der Name des Menüpunktes. Der zweite ist eine XML-Node, also ein Unterknoten, welcher die Menüeinträge enthält.

6. Speichern Sie den Flashfilm in demselben Verzeichnis wie die XML-Datei menubar.xml.

7. Sie können jetzt den Flashfilm testen (Strg-Enter). Wie zu sehen, ist das Menü ist schon bedienbar. Das Einzige was fehlt, ist eine Ereignisbehandlung wenn ein Menüpunkt ausgewählt wird.

Jetzt wird noch eine Ereignisbehandlungsroutine erstellt. Mit wird ein gewählter Menupunkt aufgefangen und eine Aktion ausgeführt. Zum Beispiel kann eine Funktion aufgerufen werden.

8. Fügen Sie folgenden Code (Listing 5.3.9) an den zuvor erstellten ActionScript-Code aus Listing 5.3.8. Danach können Sie den Flashfilm ausführen (Strg+Enter).

```
18
19 var menuListener = new Object();
20 menuListener.change = function(evt) {
21    var menu = evt.menu;
22    var item = evt.menuItem;
23    switch(item) {
24       case menu.datei_oeffnen:    trace("Punkt gewählt");
25                                   break;
26    }
27 }
28
29 myMenu.addEventListener("change", menuListener);
```

Listing 5.3.9: Erweiterung um eine Ereignisbehandlungsroutine

Ein Listener-Objekt wird erstellt (Zeile 19) und mit einer Funktion verbunden die aufgerufen wird wenn der Listener aktiv ist. Dieser wird in Zeile 29 aktiviert. Dort wird der Listener an das Menü myMenu angehangen. In den Zeilen 21 bis 27 wird der Menüaufruf behandelt.

Die Variable `menu` referenziert den Menüpunkt und die Variable `item` den entsprechenden Eintrag.

In der Fallunterscheidung per `switch` werden die Einträge abgefangen und für die für die einzelnen Menüpunkte die Aktionen ausgeführt. Aus dem Menüeintrag wird jeweils der passende Instanzname, der in der XML-Datei vergeben wurde, übergeben und per `case` eine Aktion ausgeführt. Wenn jetzt weitere Menüpunkte oder Einträge in die XML-Datei integriert werden, so muss hier für jeden Menüeintrag der Instanzname angegeben werden und eine Aktion die darauf folgen soll.

Ein default-Eintrag gibt es nicht, da kein anderer Fall eintreten kann als die Auswahl eines Menüpunktes.

Genau hier ist ein ganz großes Manko.

Das Problem

Die XML-Datei lässt sich zwar erweitern und das Menü wird auch dargestellt, nur abgefangen werden die ausgewählten Menüpunkte nicht. Für jeden Punkt muss die `switch`-Verzweigung angepasst werden.

Für die dynamische Anwendung ist das aber nicht machbar. Für jeden Themeneintrag wird ein Menüpunkt erstellt. Da dies aber im Nachhinein geschieht, müsste der Administrator des Tools jedes Mal die Flashdatei verändern.

Da aber in der Literatur und im Internet die Vorgehensweise der obigen entspricht und es keine alternativen befriedigenden Lösungen gibt, die das Problem, das hier vorliegt, lösen, musste der Autor sich etwas einfallen lassen. Nach einiger weiterer Recherchen, die zu keiner Lösung führten, kam die Idee, sich die referenzierten Variablen genau anzuschauen. Fügt man zwischen den Zeilen 22 und 23 aus Listing 5.3.9 folgende Zeile ein, so wird bei einer Auswahl eines Menüpunktes der Inhalt der Variablen `item` im Ausgabefenster angezeigt.

```
trace(item);
```

In der Ausgabe erscheint eine XML-Node.

```
<menuItem label="Öffnen" instanceName="datei_oeffnen" />
```

Was liegt also näher als diese Node abzufragen.

```
var menuListener = new Object();
menuListener.change = function(evt) {
    var item = evt.menuItem;
    trace ("Label: " + item.attributes.label + "\nInstanzname: " +
       item.attributes.instanceName +
       "\n---------------------------------------------");
}
```

Listing 5.3.10: "Die" Lösung

Fügt man den Code aus Listing 5.3.10 statt der Zeilen 19 bis 27 aus Listing 5.3.9 ein, so werden das Label und der Instanzname in der Ausgabe ausgegeben (Abbildung 5.3.7).

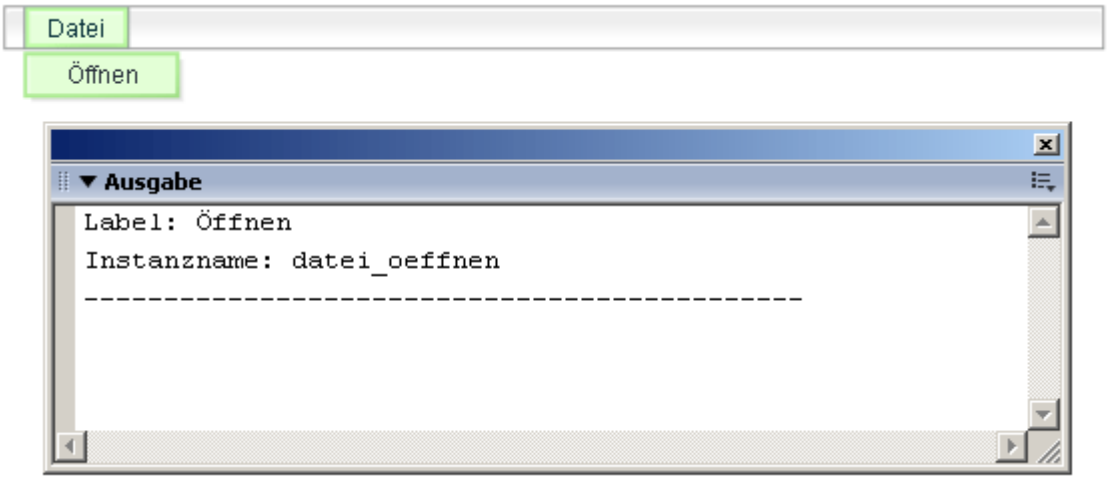


Abbildung 5.3.7: Ausgabe in Flash

Jetzt kann eine Funktion programmiert werden die als Argument das `label` oder `instanceName` bekommt und dieses auswerten kann und weiter verzweigt.

Die XML-Node der Datei menubar.xml die das Tag `<menuItem>` enthält kann auch erweitert werden.

```
<menuItem label="Öffnen" instanceName="datei_oeffnen"
 sprungZu="Thema"/>
```

Mit einer Funktion, zum Beispiel `tuwas(item.attributes.sprungZu)` könnte eine entsprechende Aktion ausgeführt werden.

In dem E-Learning-Tool wird so ein Thema angesprungen und das anhand der Überschrift des Themas.

Zu jedem Thema in der XML-Datei hauptdatei.xml gibt es einen entsprechenden Eintrag für einen Menüpunkt.

```
<menuBar menu="Einweisung" submenu="E2" menuItem="Anwendungen"/>
```

Da aber nicht die ganze XML-Datei in das XML-Objekt für das Menü gespeichert werden kann, weil die Datei nicht dem Aufbau der XML-Struktur für die Menüleiste entspricht, müssen die Menüeinträge der Themen herausgefiltert werden und ein entsprechendes XML-Objekt für die Menüleiste dynamisch generiert werden.

XML-Objekte dynamisch mit Inhalt füllen

In dem E-Learnig-Tool werden alle `<menuBar…/>`-Einträge aus der XML-Haupt-Datei "hauptdatei.xml" durchlaufen und die einzelnen Werte in Arrays, die den Menüpunkten entsprechen, erkennbar am Attribut `menu`, gespeichert. Danach werden die Inhalte der Arrays sortiert. Hier (Abbildung 5.3.8) ein beispielhaftes Array.

submenu	menuItem
	Punkt A
Menü B	Punkt B.1
Menü B	Punkt B.2
Menü B	Punkt B.3
Menü C	Punkt C.1
Menü C	Punkt C.2

Abbildung 5.3.8: Beispielarray

Jedes einzelne Array ist für einen Menüpunkt zuständig. Gehen wir davon aus, dass das Array aus Abbildung 5.3.8 zu einem Menüpunkt gehört. Es gibt eine For-Schleife in der Anwendung die das fertig sortierte Array in ein XML-Objekt für die Menüleiste schreibt. Die Menüpunkte werden so erstellt, wie sie in dem Array stehen.

Wo unter submenu kein Eintrag ist, wird der Eintrag unter menuItem direkt in das Menü als anwählbaren Menüpunkt eingetragen. Ist ein Eintrag unter submenu, so wird ein Untermenüpunkt mit diesem Namen erstellt und der Eintrag unter menuItem als Menüpunkt in ihm platziert. Sind weitere Menüpunkte zu diesem Untermenü, werden diese zu dem jeweiligen Untermenü dazugefügt.

Um Inhalt für ein XML-Objekt zu erzeugen muss, nachdem das Objekt selber erzeugt wurde, als allererstes ein Wurzelknoten erstellt werden.

```
var myMenuBar = new XML(); // XML-Objekt erzeugen

myMenuBar.contentType = "text/xml"; // Typ festlegen
```

Zur Erstellung von XML-Elementen gibt es in dem E-Learning-Tool zwei Methoden der XML-Klasse, die Methode `createElement("…")` und `appendChild(…)`.

```
wurzelKnoten = myMenuBar.createElement("menuBar");
```

Erstellt das XML-Element `menuBar` und speichert es in `wurzelKnoten`. Dieses wird mit folgender Zeile in das XML-Objekt myMenuBar angehangen.

```
myMenuBar.appendChild(wurzelKnoten); //Wurzelknoten erstellen
```

Um nun weitere Elemente in den so erstellten Wurzelknoten einzufügen, wird wie folgt vorgegangen.

```
datensatzKnoten = myMenuBar.createElement("menu");
datensatzKnoten.attributes.listener = "menuListener";
datensatzKnoten.attributes.instanceName = "Einweisung";
datensatzKnoten.attributes.label = "Einweisung";
wurzelKnoten.appendChild(datensatzKnoten);
```

Es wird das Element `menu` erstellt, in dem die drei Attribute listener, instanceName und label mit den jeweiligen Inhalten menuListener und zweimal Einweisung eingetragen werden. Diese werden per appendChild in den Wurzelknoten eingefügt. So erhalten wir das XML-Objekt mit folgendem Inhalt (Listing 5.3.11).

```
<menuBar>
   <menu listener="menuListener" instanceName="Einweisung" label="Einweisung"/>
</menuBar>
```

Listing 5.3.11: Erstelltes XML-Objekt myMenuBar

Genauso werden die jeweiligen Tags `<menuItem>`, in der Art wie vorher mit dem Beispielarray beschrieben (siehe auch Abbildung 5.3.8), erzeugt. Es werden zwei Arrays bearbeitet, das Array menuEinweisung und menuTheorie, welche den beiden Hauptmenüpunkten Einweisung und Theorie entsprechen.

Zusätzlich wird noch ein drittes Menü Theorie erzeugt, welches allerdings fest im Code steht und nicht von außen beeinflussbar ist, da hier feste Bestandteile der Anwendung integriert sind. Diese sind der Aufruf des Glossarfensters ohne auf einen Link zu klicken und weitere Punkte, wie zum Beispiel das Impressum.

Auf diese Weise wird das komplette Menü dynamisch erzeugt.

Glossarfenster – UI Komponente Window

Bei Ablauf des E-Learning-Tools gibt es zwei Möglichkeiten, das Glossarfenster aufzurufen. Einmal über Links, die im Text dargestellt werden und die andere Möglichkeit ist über den Menüpunkt Extras/Glossar zu erreichen.

Die UI Komponente Window muss in der Bibliothek enthalten sein. Wird die Komponente auf der Bühne platziert, kann sie auch gleich wieder von ihr gelöscht werden. So ist sie aber noch in der Bibliothek, wo sie per ActionScript in der Anwendung aufgerufen und dargestellt wird.

Der Inhalt des Glossarfensters wird mit einem Movieclip gefüllt. Bei diesem müssen über die Bibliothek die Eigenschaften geändert werden. Der Movieclip muss für den Export in ActionScript freigegeben werden.

```
var win = new Object();
win = mx.managers.PopUpManager.createPopUp(_root, mx.containers.Window, true,
      {title: "Fenster", contentPath: "fenster_inhalt", closeButton: true });

var btnClose = new Object();
btnClose.click = function() {
    win.deletePopUp();
};
win.addEventListener("click", btnClose);
win.setSize(400, 200);
win.move(100, 100);
```

Listing 5.3.12: Erstellung eines Fensters

Es wird eine Objektvariable `win` erzeugt und mit der Methode `createPopUp(…)` ein Fenster erzeugt. In Anführungszeichen hinter `contentPath:` steht der für ActionScript freigegebene Movieclip-Bezeichner. Dadurch wird der Movieclip mit dem Fenster verknüpft.

In den folgenden Zeilen wird eine Ereignisbehandlung für den Schliessen-Button erstellt, der mit `closeButton: true` eingebunden wurde. Dieser schließt das Fenster mit `win.deletePopUp()`. Anschließend wird der Listener gestartet und die Größe und Position des Fensters festgelegt.

In dem E-Learnig-Tool wird im Glossarfenster außer dem Fenstertitel, dem Titel des Schlagwortes und dem Text mit der Erklärung noch eine Liste mit allen Glossareinträgen angezeigt. Diese Liste wird mit der UI Komponente List erstellt.

Glossarfenster – UI Komponente List

Eine Liste mit allen Glossareinträgen kann unter Umständen sehr lang werden. Wie kann diese in das Glossarfenster untergebracht werden? Flash bietet für Listen eine UI Komponente an, die Komponente List. Es wird eine Box dargestellt, in der alle Einträge untereinander aufgelistet werden. Mit einem Scrollbalken kann jetzt durch alle Einträge gescrollt werden. Diese lassen sich einzeln Anklicken, woraufhin intern ein Listener die Aktion abfängt und zum Beispiel der Erklärungstext dargstellt wird.

```
for (var i:Number = 0; i < _root.GlossarEintraege.length; i++) {
 glossarList.addItem(_root.GlossarEintraege[i][0],
_root.GlossarEintraege[i][1]);
}
```

Listing 5.3.13: Array der Glossarschlagworte in die List-Komponente hinzufügen

Die Glossarschlagworte, die in einem Array gespeichert wurden (siehe Abbildung 5.1.3), werden der UI Komponente List mit der Methode `addItem(…)` hinzugefügt (Listing 5.3.13). Das erste Argument ist das Schlagwort selber, das zweite die Stelle im XML-Objekt, an dem es zu finden ist.

Die Ereignisbehandlungsroutine, die hier direkt im List-Objekt per `on(change){…}` abgefragt wird, aktualisiert den Glossartext und die Glossarüberschrift.

Historynavigation

Die Historynavigation ist eine Navigation, die so arbeiten soll, wie die vom Internet Explorer. Wie arbeitet aber eine derartige Navigation?

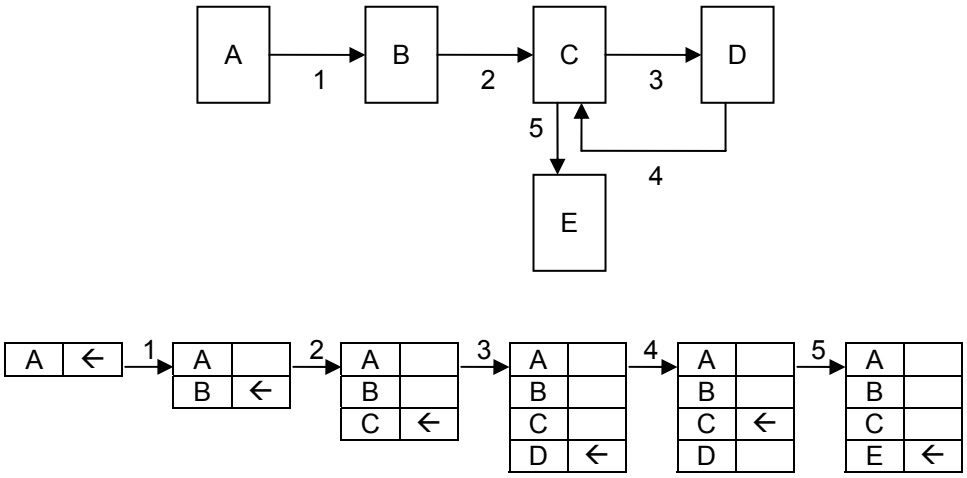


Abbildung 5.3.9: Arbeitsweise einer Hisorynavigation

In Abbildung 5.3.9 ist die Arbeitsweise einer Historynavigation zu sehen.

Die einzelnen Kästchen mit den Buchstaben stellen besuchte Seiten dar. Die untere Reihe stellt das Array mit einem Zeiger, hier als Pfeil, dar. Als allererstes ist natürlich die erste Seite zu merken. Es wird A in dem Array gespeichert und der Zeiger auf die Stelle gesetzt. Mit Besuch der Seite B wird das Array erweitert und der Zeiger auf die aktuelle Position gesetzt. Bis zum dritten Seitenwechsel (3) ist das Array auf vier Elemente gewachsen. Es wird bei (4) einmal in der History zurückgesprungen, hier wird nur der Zeiger verändert. Daraufhin wird eine neue Seite E aufgerufen (5). Das Array wird an der vierten Stelle gelöscht und durch die neue Seite ersetzt. Der Zeiger wird auf die Seite gesetzt.

Wenn jetzt beispielsweise in der History komplett zurückgegangen wird, also auf die erste Stelle im Array (Seite A, Zeiger auf A), so bleiben die Elemente erhalten. Wird von dort aus eine neue Seite angesprungen ohne die History zu benutzen, so wird auch wenn es die Seite B sein sollte, die History bis dahin gelöscht und der Zeiger auf die aktuelle Seite gesetzt. Es sind jetzt nur noch die Seite A und die neue Besuchte Seite in Array gespeichert.

6 Installation und Anleitung

Die Benutzung des E-Learning-Tools im Internet wird sich, wenn das Tool im Hahn-Meitner-Institut Berlin GmbH eingesetzt wird, in der Hinsicht ändern, dass sie auf einer anderen Domain platziert wird. Damit ändern sich dann auch die Zugangsdaten.

Für diese Diplomarbeit wird die Anwendung für das Internet aber auf der Internetseite http://diplom.creator4you.com gelagert sein.

Im Folgenden wird die Funktionsweise des E-Learnig-Tools erklärt.

6.1 Benutzerhandbuch für den Anwender

Diese Version der Anleitung bezieht sich auf die fertige CD-ROM-Version (Kapitel 9.5).

Herzlich Willkommen zur Einweisung an den Experimenten am Hahn-Meitner-Institut Berlin GmbH. Mit diesem Tool können Sie sich über die einzelnen Experimentaufbauten in der Experimentierhalle informieren.

Start der Anwendung aus dem Internet

Voraussetzung zur Benutzung im Internet ist ein Internetfähiger PC (MAC oder PC mit mindestens 800MHz). Falls Sie kein Flash-Plugin der Version 7 für Ihren Browser installiert haben, so können Sie sich das Plugin (erhältlich für Netscape, Mozilla, Opera, und CompuServe) in der neuesten Version von der Seite http://www.macromedia.com downloaden und installieren.

Nach einer erfolgreichen Installation rufen Sie die Seite http://diplom.creator4you.com auf. Sie werden nach einem Benutzername und einem Passwort gefragt.

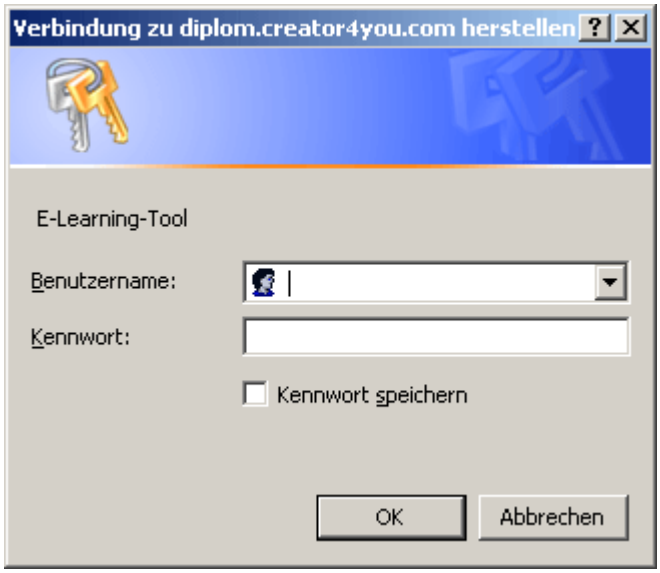


Abbildung 6.1.1: Passwortabfrage zur Benutzung im Internet

Geben Sie folgende Daten ein:

Benutername: diplom

Passwort: ws2004

Start der Anwendung von CD-ROM

Voraussetzung ist ein PC mit mindestens 800 MHz und installiertem Windows Betriebssystem (Win98/ME/2000/2003/XP).

Legen Sie die CD-ROM in das CD-ROM-Laufwerk: Die Anwendung sollte automatisch starten. Startet sie nicht automatisch, so öffnen Sie das Verzeichnis der CD und mach Sie einen Doppelklick auf die Datei start.exe.

Die Oberfläche

Die eigentliche Oberfläche der Anwendung ist nicht davon abhängig, ob sie aus dem Internet oder von der CD gestartet wird. Hier werden nur die Oberflächenelemente der Anwendung erklärt.

Nach dem Aufruf bietet sich Ihnen folgende Oberfläche:

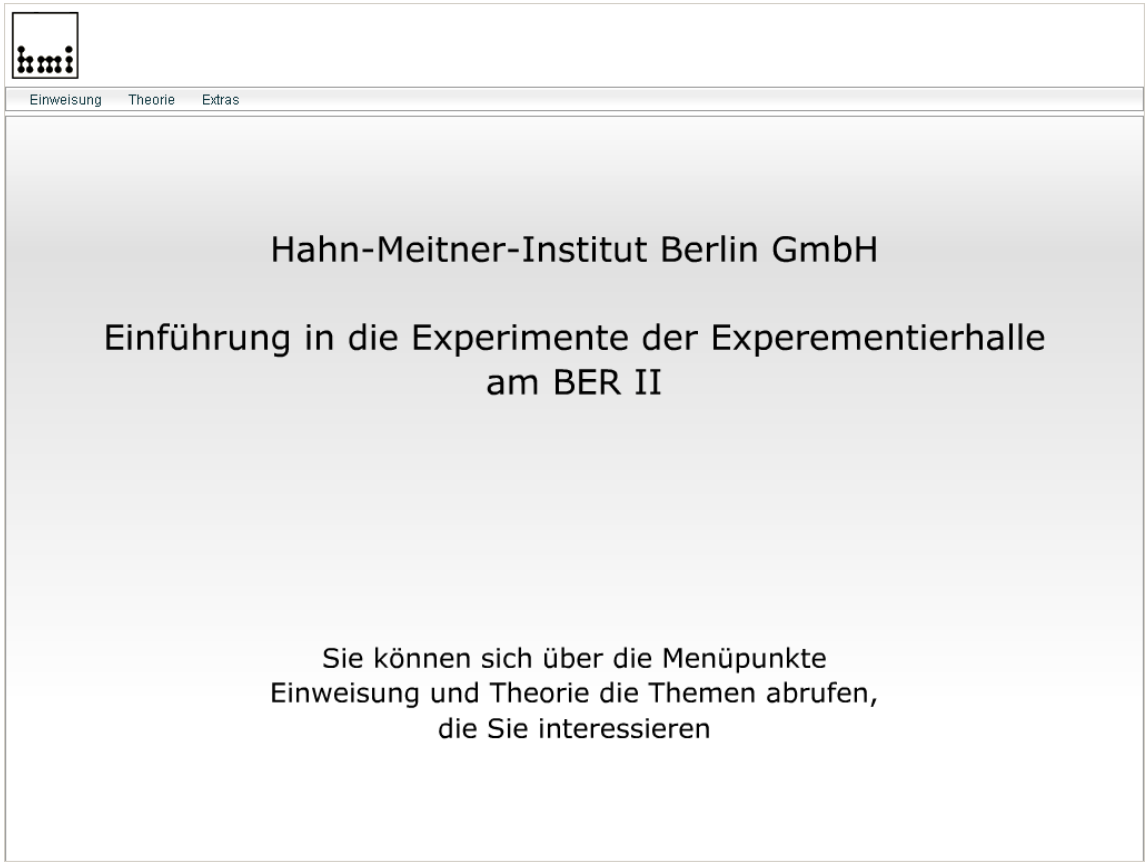


Abbildung 6.1.2: Oberfläche nach dem Start

Sie können jetzt in der Menüleiste einen der Menüpunkte Einweisung, Theorie oder Extras auswählen.

Wenn Sie sich einige Themen angeschaut haben, könnte die Oberfläche so aussehen:

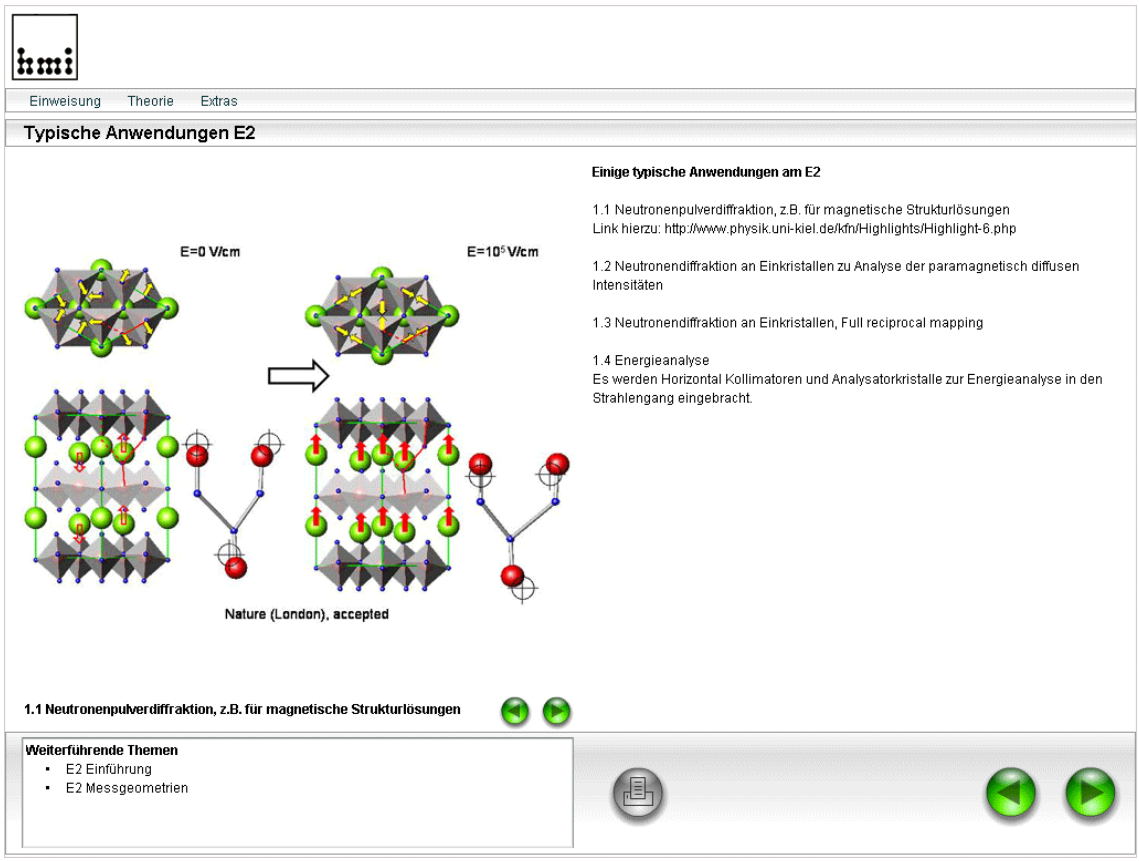


Abbildung 6.1.3

Nachfolgend werden die einzelnen Elemente der Oberfläche anhand der Abbildung 6.1.3 erklärt.

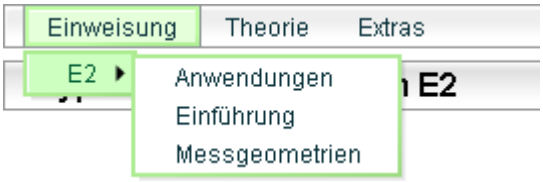


Abbildung 6.1.4: Die Menüleiste

In der Menüleiste können Sie zu dem jeweiligen Menüpunkt navigieren, der Sie interessiert. Im Menü Extras finden Sie das Glossar, aber dazu später mehr.

Unter der Menüleiste wird die Überschrift zu einem gewählten Thema angezeigt. Darunter befindet sich der Bereich, in dem die Mediendateien, wie Bilder und Videos dargestellt werden. Jedes Bild hat eine eigene Beschreibung und rechts daneben eine Navigation für die einzelnen Bilder zu einem Thema.

Abbildung 6.1.5: Bildbeschreibung und Navigation

Die Navigation erscheint nur, wenn es mehrere Mediendateien zu einem Thema gibt. Steht nur ein Bild zur Verfügung, so wird die Navigation ausgeblendet.

Unter der Bildbeschreibung befindet sich ein Feld für die Anzeige von verwandten Themen. Diese lassen sich anklicken, woraufhin das Thema angezeigt wird (Abbildung 6.1.6).

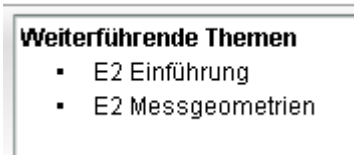


Abbildung 6.1.6: Links zu weiteren Themen

Rechts neben dem angezeigten Bild, steht der dem Thema zugehörige Text. Sind einige Wörter blau markiert, so können diese angeklickt werden. Daraufhin öffnet sich das Glossarfenster mit dem jeweiligen Begriff, der erklärt wird (Abbildung 6.1.7).

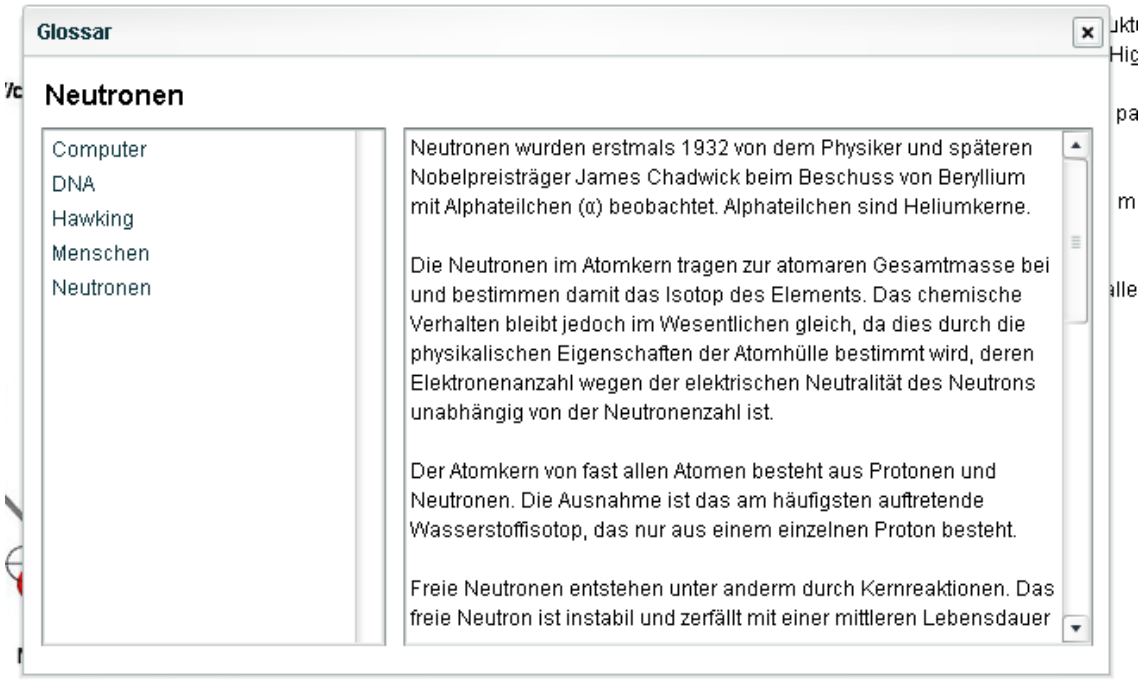

Glossar
Neutronen
Computer
DNA
Hawking
Menschen
Neutronen
Neutronen wurden erstmals 1932 von dem Physiker und späteren Nobelpreisträger James Chadwick beim Beschuss von Beryllium mit Alphateilchen (α) beobachtet. Alphateilchen sind Heliumkerne.
Die Neutronen im Atomkern tragen zur atomaren Gesamtmasse bei und bestimmen damit das Isotop des Elements. Das chemische Verhalten bleibt jedoch im Wesentlichen gleich, da dies durch die physikalischen Eigenschaften der Atomhülle bestimmt wird, deren Elektronenanzahl wegen der elektrischen Neutralität des Neutrons unabhängig von der Neutronenzahl ist.
Der Atomkern von fast allen Atomen besteht aus Protonen und Neutronen. Die Ausnahme ist das am häufigsten auftretende Wasserstoffisotop, das nur aus einem einzelnen Proton besteht.
Freie Neutronen entstehen unter anderm durch Kernreaktionen. Das freie Neutron ist instabil und zerfällt mit einer mittleren Lebensdauer

Abbildung 6.1.7: Das Glossarfenster

In dem Glossarfenster wird der jeweilige Begriff, der aus dem Text heraus verlinkt wurde, erklärt. Über der Auswahlliste, aus der Sie sich auch Begriffe auswählen können, wird das zu erklärende Wort dargestellt. In der Liste stehen alle verfügbaren Glossarbegriffe. Durch diese können Sie, falls es mehr gibt als auf der sichtbaren Liste erscheinen, mittels eines Scrollbalkens durchscrollen. Auf der rechten Seite ist der Text, der die Erklärung zu dem Begriff enthält. Schließen können Sie das Glossarfenster mit dem Kreuz in der Fensterleiste.

Abbildung 6.1.8: Schließbutton für das Glossarfenster

In der Abbildung 6.1.9 sehen Sie noch eine Navigationsleiste. Diese ist gleich neben den weiteren Themen angesiedelt.

Abbildung 6.1.9: Navigationselemente neben den weiteren Themen

Der graue Button ist eine Funktion, die später in einer weiteren Version der Anwendung nutzbar ist. Mit diesem Button können Sie dann drucken.

Rechts daneben finden Sie zwei Pfeile, die der Historynavigation zugeordnet sind. Diese arbeitet so wie die des Internet Explorers. Rufen Sie mehrere Themen nacheinander ab, so wird ein Button zum Zurückschalten in ein vorher besuchtes Thema eingeblendet. Gehen Sie ein Schritt zurück, wird auch ein Button zum wieder Vorwärtsschalten eingeblendet. Wenn Sie ein neues Thema aus dem Menü oder über die weiteren Themen anwählen, wird der Button zum Vorschalten ausgeblendet.

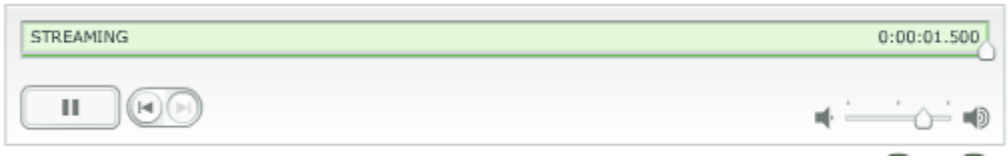


Abbildung 6.1.10: Navigationsleiste bei der Videodarstellung einer Mediendatei

Zu manchen Themen gibt es ein Video und es erscheint eine Navigationsleiste wie in Abbildung 6.1.10. Hier können Sie das Video steuern, zum Beispiel auf Pause drücken oder aber auch den Ton, wenn vorhanden, leiser oder ganz aus machen.

Und nun viel Spaß mit der Anwendung.

6.2 Benutzerhandbuch für den Administrator

Diese Version der Anleitung bezieht sich auf die CD-ROM Diplom (Kapitel 9.4).

Voraussetzung ist ein PC (Mac oder PC mit mindestens 800MHz). Für die Bearbeitung der Dateien wird ein UTF-8 fähiger Texteditor benötigt. Der Standarteditor von Windows XP SP1, das Notepad, kann dieses.

Für die Administration der Anwendung müssen die zu bearbeitenden Dateien auf dem Computer gespeichert werden. Dazu kopieren Sie die Daten aus dem Verzeichnis "Anwendung CD-ROM" komplett von der beiliegenden CD-ROM in ein freies Verzeichnis auf ihren Rechner. Danach müssen Sie noch den Schreibschutz der Dateien entfernen. Gehen Sie dazu auf das Verzeichnis, in dem die kopierten Daten liegen und drücken die rechte Maustaste. Dort gehen Sie auf die Eigenschaften und entfernen den Schreibschutz.

Es gibt drei Verzeichnisse, das media-Verzeichnis, das texte-Verzeichnis und das xml-Verzeichnis. In diesen befinden sich die zu ändernden Dateien.

Neues Thema anhand eines Beispiels dazufügen

Die zugehörigen Bilddateien und Texte finden Sie auf der CD-ROM unter Beipieldateien/Bilder und Texte.

Gehen wir davon aus, dass Sie folgenden Eintrag in die Anwendung einpflegen möchten:

Menüpunkt: Theorie → Test →Punkt 3

Überschrift: Punkt 3 Test

Unterüberschrift: Punkt 3 Unterüberberschrift

Bild: punkt3.jpg

Bildbeschreibung: Dies ist eine Bildbeschriftung

Zugehöriger Text: punkt3.txt

weitere Themen: Punkt 1 Test und Punkt 2 Test

Öffnen Sie dazu die Datei hauptdatei.xml im Unterverzeichnis xml mit einem UTF-8 fähigen Texteditor. Unter Windows XP SP1 können Sie den Editor unter Programme/Zubehör nehmen. Im Öffnen-Dialog müssen Sie den Dateityp auf "Alle Dateien" stellen.

Sie bekommen jetzt die XML-datei zu sehen. Gehen Sie an das Ende der Datei und fügen Sie vor dem Tag </main> folgende Zeilen ein.

```
<eintrag>
  <ueberschrift>Punkt 3 Test</ueberschrift>
  <unterueberschrift>Punkt 3 Unterüberschrift</unterueberschrift>
  <media>
     <eintrag src="punkt3.jpg">Dies ist eine Bildbeschriftung</eintrag>
  </media>
  <text>punkt3.txt</text>
  <weitereThemen>
     <eintrag>Punkt 1 Test</eintrag>
     <eintrag>Punkt 2 Test</eintrag>
  </weitereThemen>
  <menuBar menu="Theorie" submenu="Test" menuItem="Punkt 3"/>
  <index>
     <eintrag></eintrag>
  </index>
</eintrag>
```

Speichern Sie die Datei.

Gehen Sie in das Verzeichnis "Beispieldateien/Bilder und Texte" auf der CD-ROM und kopieren Sie die Datei "punkt3.jpg" in das Verzeichnis media. Die Datei "punkt3.txt" kopieren Sie in das Verzeichnis texte.

Nun können Sie die Anwendung austesten. Sie werden feststellen, dass sich ein neuer Menüpunkt unter dem Theorie-Menü befindet. Gehen Sie auf Ihren erstellten Untermenüpunkt Test und klicken Sie auf Punkt 3.

Wichtig! Die Überschrift unter <ueberschrift> darf nur einmal in der Datei vorkommen. Anhand dieser wird das Menü ausgewertet.

Wichtig! Unter <weitereThemen> dürfen nur Überschriften stehen, die im XML-Dokument schon unter <ueberschrift> vorkommen.

Eigene Texte

Eine eigene Textdatei verfassen Sie folgendermaßen. Öffnen Sie den Editor und schreiben Sie folgenden Text.

var1=

Dieses ist eine Variable. Hinter dem Gleichzeichen können Sie ganz normal Ihren Text eingeben. Sie können griechische Sonderzeichen eingeben, wie sie in TEX verwendet werden. Eine Liste der vorgegebenen Sonderzeichen finden Sie im Anhang 9.1. Diese sind in der Datei sonderzeichen.xml im xml-Verzeichnis vorgegeben. Diese Datei kann auch bearbeitet werden. Aber dazu später mehr.

Speichern können Sie den Text unter "Datei/Speichern Unter". Stellen Sie den Dateityp auf "Alle Dateien" und die Codierung auf "UTF-8". Gehen Sie in das Verzeichnis texte, geben als Dateinamen "punkt3.txt" ein und speichern.

Eigene Bilder

Eigene Bilder müssen im JPG-Format gespeichert werden und müssen eine Auflösung von 500x480 Pixel besitzen.

Glossareinträge hinzufügen

Um Glossareinträge hinzuzufügen muss die Datei "glossar.xml" aus dem Verzeichnis xml editiert werden. Gehen Sie genauso vor wie bei den Themen.

Öffnen Sie die Datei mit dem Editor (Dateityp nicht vergessen).

Vor dem letzten Eintrag </glossar> geben Sie folgendes Grundgerüst ein:

```
<eintrag>
   <Schlagwort>ABC</Schlagwort>
   <text>ABC</text>
</eintrag>
```

Die Stellen an denen ABC steht, können Sie füllen. Als Schlagwort sollten sie kurze Wörter benutzen. Hier werden allerdings keine Sonderzeichen ausgetauscht.

Wichtig! Ein Schlagwort darf nur einmal in der XML-Datei als Schlagwort benutzt werden.

Speichern Sie die Datei.

Sonderzeichen hinzufügen

Gehen Sie beim Öffnen der Datei "sonderzeichen.xml" genauso vor wie bei den Glossareinträgen.

Vor dem letzten Eintrag </sonderzeichen> geben Sie folgendes Grundgerüst ein:

```
<zeichen>
   <tex>\ABC</tex>
   <unicode>ABC</unicode>
</zeichen>
```

Die Stellen, an denen ABC steht, können Sie füllen. Lassen Sie sich eine Abkürzung einfallen oder nehmen Sie eine aus TEX. Öffnen Sie die Zeichentabelle unter Programme/Zubehör/Systemprogramme/. Suchen Sie sich ein Sonderzeichen aus und kopieren es an die Stelle an der bei <unicode> ABC steht.

Wichtig! Ein <tex>-Element darf nur einmal in der XML-Datei als Schlagwort benutzt werden. Vor ihm muss ein Backslash "\" stehen.

Speichern Sie die Datei.

Videos für die Anwendung erstellen

Grundvoraussetzung ist Macromedia Flash MX 2004 Professional.

Für Videos brauchen Sie zwei Dateien, eine SWF-Datei (Flash) und eine FLV-Datei (Flash-Video).

Video konvertieren

Öffnen Sie Flash und erstellen ein neues Flash-Dokument (Datei→Neu).

Drücken Sie die Taste F11. Es erscheint rechts die Bibliothek für dieses Dokument.

Unter "Datei→Importieren→In Bibliothek Importieren…" können Sie gängige Formate für Videos in die Bibliothek importieren.

1. Es erscheint ein Dialog.

2. Gehen Sie auf "Gesamtes Video importieren" und auf Weiter.

3. Auf der nächsten Seite können Sie alle Einstellungen so belassen.

4. Gehen Sie auf Beenden.

Das Video sollte jetzt importiert werden und ist danach in der Bibliothek zu finden.

1. Klicken Sie mit der rechten Maustaste auf das Video in der Bibliothek.

2. Gehen Sie auf Eigenschaften und drücken Exportieren.

3. Speichern Sie das Video in einem temporären Verzeichnis.

Das Video wird im FLV-Format gespeichert.

Video in eine SWF-Datei einbinden

Öffnen Sie Flash und öffnen die Datei "video.fla" aus dem Verzeichnis "Beispieldateien/Bilder und Texte" von der CD-ROM.

Speichern Sie die Datei gleich wieder mit Datei→Speichern unter in dem temporären Verzeichnis wo das FLV-Video ist.

Öffnen Sie den Komponenten-Inspektor

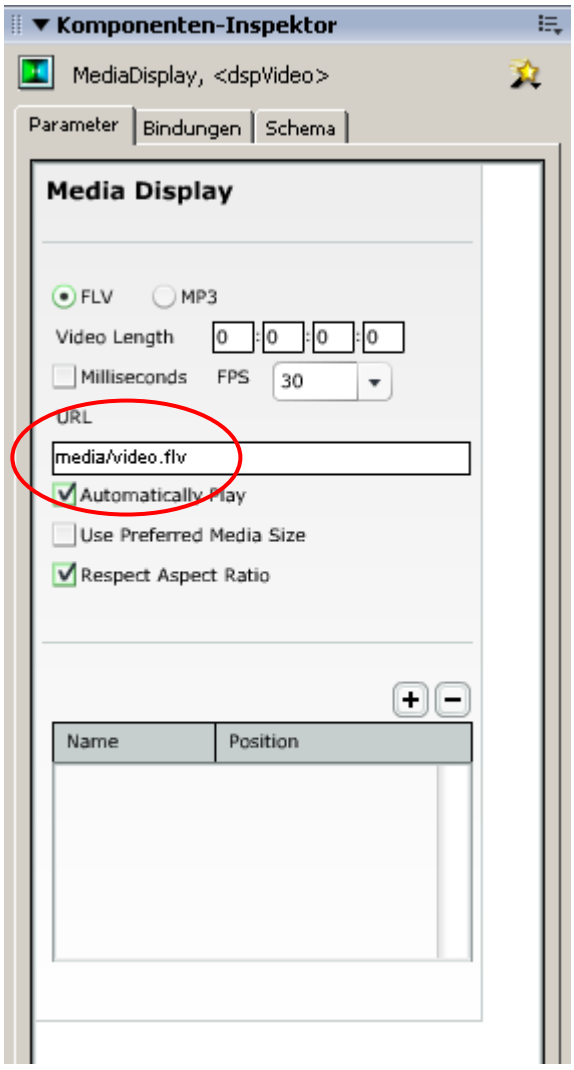


Abbildung 6.2.1: Der Komponenten-Inspektor für MediaDisplay unter Flash

Falls Sie diesen Inspektor nicht vorfinden, klicken Sie auf der Bühne auf die weiße Fläche über der Videosteuerung. Geben Sie statt media/video.flv hinter media/ den Dateinamen Ihres Videos an (Abbildung 6.2.1).

Drücken Sie Strg + Enter. Falls Sie das Video nicht sehen, haben Sie keine Angst, es ist alles in Ordnung. Beenden Sie jetzt Flash.

Kopieren Sie die erstellte SWF-Datei und FLV-Datei, aus dem temporären Verzeichnis, in das Verzeichnis media. Jetzt müssen Sie in der hauptdatei.xml statt einer JPG-Datei unter <eintrag src="punkt3.jpg">, die neu erstellte SWF-Datei angeben.

Fertig.

Flashfilme (SWF-Datei einbinden)

Als Vorlagendatei können Sie die Datei "Flashvorlage.swf" aus dem Verzeichnis "Beispieldateien/Texte und Bilder" nehmen. Die Größe der Bühne muss 500x480 Pixel betragen. Den fertigen Flashfilm, also die SWF-Datei, binden Sie so ein wie ein Video.

Kopieren Sie die Datei in das Verzeichnis media und tragen Sie den Dateinamen in die XML-Dartei "hauptdatei.xml", an der Stelle <eintrag src="punkt3.jpg"> hinter src, ein.

Erstellung einer neuen CD-ROM

Da die Dateien jetzt im Verzeichnis aktualisiert sind, können Sie den Inhalt des Verzeichnisses, in welchem Sie vorher die CD-ROM kopiert haben, wieder auf eine CD-R brennen.

Dateien im Internet

Kopieren Sie die Dateien aus dem Verzeichnis "Anwendung Internet" per FTP auf den Webspace Ihres Anbieters. Wenn die Rechte richtig gesetzt sind, sollte die Anwendung über die Datei "index.html" starten.

Die Änderung der Inhalte erfolgt genau wie oben beschrieben. Kopieren Sie nur die aktualisierten Dateien per FTP auf Ihren Webauftritt zurück.

7 Fazit und Ausblick

Fazit

Erste positive Erfahrungen mit Flash hatte ich in meinem Studium ja schon gemacht. Mit der neuen Version von Flash, also Flash MX 2004 Professional, hat sich auch die Oberfläche geändert. Nach einer gewissen Einarbeitungszeit, die mit dem Projekt angefangen hat, habe ich mich dann auch zurechtgefunden.

Die einzelnen UI Komponenten in der neuen Version haben eine Menge Arbeit gemacht. Doch nach viel herumprobieren und recherchieren habe ich diese auch in den Griff bekommen. Einzig die Menükomponente hat mir Kopfzerbrechen bereitet.

Zudem kommen noch einige Kleinigkeiten in der ActionScript-Programmierung, die ich dazulernen musste. Ich frage mich allerdings bis heute, warum das mit den Klassen für XML nicht funktioniert hatte. Das ist ein Problem, dem ich mich weiterhin widmen werde.

Mit dem E-Learning-Tool ist ein kleines Mini-CMS entstanden. Der Aufwand der dafür betrieben wurde ist allerdings nicht ohne. Es sind bis zum vollständigen Projektabschluss (siehe Abbildung 2.1.1) noch einige Hürden zu nehmen.

Das Eingabeformular stellt eine große Aufgabe dar. Dieses Formular soll vorab die komplette vorhandene XML-Struktur der Hauptdatei einlesen. Damit kann dann eine Überprüfung der eingegebenen Daten, zum Beispiel ob ein Punkt doppelt vorkommt, dem Administrator abgenommen werden.

Ein großes Problem dabei stellt der Upload einer Datei dar, da Flash hierfür keinen Dialog bereitstellt. Einige Ansätze gibt es schon. Diese arbeiten aber im Browser mit Frames und JavaScript. Was aber keine zufrieden stellende Lösung ist.

Für den Druck auf Papier gilt das gleiche. Flash unterstützt zwar das Drucken, nur bleibt einem nichts anderes übrig, als ein festes selber vordefiniertes Format zu nehmen.

Es bleibt abzuwarten, was die nächste Version von Flash bringt.

Ausblick

Die Anwendung soll am Hahn-Meitner-Institut Berlin GmbH vorerst für die Experimente E2, E2a, E3 und E7 verwendet werden. Nach einer erfolgreichen Testphase soll das Produkt dann um weitere Experimentaufbauten erweitert werden.

In der nächsten Version der Anwendung werden dann hoffentlich alle geplanten Funktionen zur Verfügung stehen. Oben im Fazit wurde schon erwähnt, dass einige Funktionen noch nicht von Flash unterstützt werden. Aber es gibt ja immer wieder Lösungen, die vorher nicht für möglich gehalten waren.

8 Literatur und Quellenangaben

Literatur

Michael Rüttger
XML mit Flash MX 2004, 1. Auflage 2004, mitp-Verlag
ISBN 3-8266-0992-1

Dirk Lois, Svend Nissen
Flash MX 2004 und ActionScript Kompendium, Data Becker 2004
ISBN 3-8272-6672-6

Elizabeth Castro
XML fürs World Wide Web, Markt & Technik 2001
ISBN 3-8272-5994-0

Internet

Wikipedia – Die freie Enzyklopedie
URL: http://de.wikipedia.org - Stand: 18.12.2004

Macromedia
URL: http://www.macromedia.com - Stand: 18.12.2004

Flashforum
URL: http://www.flashforum.de - Stand: 18.12.2004

Hahn-Meitner-Institut Berlin GmbH
URL: http://www.hmi.de – Stand: 18.12.2004

IDM Computer Solutions, Inc – UltraEdit
URL: http://www.ultraedit.com – Stand: 18.12.2004

9 Anhang

9.1 Voreingestellte Sonderzeichen der Datei "sonderzeichen.xml"

T_EX	Zeichen	T_EX	Zeichen
\alpha	α	\varsigma	ς
\beta	β	\tau	τ
\gamma	γ	\upsilon	υ
\delta	δ	\phi	φ
\epsilon	ε	\chi	χ
\zeta	ζ	\psi	ψ
\eta	η	\omega	ω
\theta	θ	\Gamma	Γ
\iota	ι	\Delta	Δ
\kappa	κ	\Theta	Θ
\lambda	λ	\Lambda	Λ
\mu	μ	\Xi	Ξ
\nu	ν	\Pi	Π
\xi	ξ	\Sigma	Σ
\omicron	ο	\Upsilon	Υ
\pi	π	\Phi	Φ
\rho	ρ	\Psi	Ψ
\sigma	σ	\Omega	Ω

Abbildung 9.1.1: Voreingestellte Sonderzeichen

9.2 Versuch ein XML-Objekt in einer Klasse zu generieren

```
class XMLLoad {
 static private var xmlObj:XML;

 static function run(Void):Void {
        trace ("run aufgerufen...");
 }

 public function handleLoad(success:Boolean):Void {
        if (success) {
               trace(this.toString());
               run();
        }
 }

 public function loadData(File:String):Void {
        xmlObj = new XML();
        xmlObj.ignoreWhite = true;
        xmlObj.contentType = "text/xml";
        xmlObj.onLoad = handleLoad;
        xmlObj.load(File);
 }
}
```

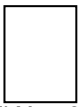
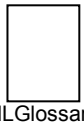
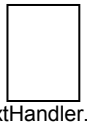
Listing 9.2.1: Klasse zur Erstellung eines XML-Objekts (Dateiname "XMLLoad.as")

```
var myObj = new XMLLoad();
myObj.loadData("sonderzeichen.xml");
```

Listing 9.2.2: Aufruf der XML-Klasse aus Listing 9.2.1

Problem: Das XML-Objekt steht nur in der Klasse zur Verfügung.

9.3 Aufbau der Anwendung (Die Dateien)

Datei	Enthält Funktionen	Beschreibung
start.swf oder start.exe	replace(suchstr:String, ersetzstr:String)	Dies ist die Hauptdatei. Sie importiert alle hier aufgeführten anderen Dateien. Sie enthält die Konfiguration und auch die Funktion zum ersetzen von Text. Auch die komplette Oberfläche ist hier enthalten.
XMLHauptdatei.as		Die XML-Datei "hauptdatei.xml" wird hier geladen. Danach wird die Funktion createXMLMenu() aufgerufen.
 XMLMenu2.as	createXMLMenu(Void):Void	Hier wird das komplette Menü erstellt und per Ereignisroutine abgefragt. Von dort aus werden die Funktionen themaSprung(thema) und glossar-Link(nummer) aufgerufen.
XMLSonderzeichen.as	tauscheSonderzeichen(Void):Void	In der Datei wird die XML-Datei "sonderzeichen.xml" geladen. Mit der Funktion tauscheSonderzeichen() werden die T_EX-Elemente in Symbole umgewandelt.
 XMLGlossar.as	glossarInArray(Void):Void markiereGlossar(Void):Void glossarLink(arg:Number):Void	Hier wird die XML-Datei "glossar.xml" geladen. glossarInArray() bereitet das Array mit den Schlagworten für das Glossarfenster vor. Das Glossarfenster wird mit der Funktion glossar-Link(Nummer) aufgerufen. markiereGlossar() markiert die Glossareinträge in der TextArea.
 textHandler.as	ladeText(file:String):Void	Hier wird der jeweilige Text zu einem Thema mit der Funktion ladeText(Datei) in die Anwendung geladen.
themaSprung.as	themaSprung(arg:String):Void	Hier ist die Funktion enthalten, die alle Felder und Movieclips aktualisiert. Der Funktion the-maSprung(Thema) wird ein Thema aus der XML-Haupt-Datei übergeben.

Dateiformate

Endung	Erklärung
FLA	Flash-Datei. Diese Datei kann mit Flash bearbeitet werden. D.h. das ist die Source-Datei.
FLV	Flash-Video Datei. Flash eigenes Format.
SWF	Fertig Kompilierter Flashfilm, kann nicht mehr bearbeitet werden. Zum abspielen wird ein Flashplayer benötigt.
AS	ActionScript-Datei. Dieses ist eine Textdatei mit ActionScript Code, der in Flash importiert wird.
JPG	Ein Komprimierungsformat der Joint Photographic Experts Group zur digitalen Speicherung von Bildern, auch JPEG genannt.
TXT	Textdatei
XML	XML-Datei

Haupt ActionScript-Code aus start.fla

```
// ********************************************************************************
// * start.fla
// * Dies ist die Hauptdatei die die anderen Module einbindet
// *
// * Autor: Andreas Tomm
// * Letzte Änderung: 20-12-2004
// ********************************************************************************

// Verzeichnisse angeben. Sollte besser extern geschehen
var src_txt:String = "texte/";
var src_media:String = "media/";
var src_xml:String = "xml/";

// Einbinden der ActionScript Dateien
#include "XMLHauptdatei.as"
#include "XMLMenue2.as"
#include "XMLSonderzeichen.as"
#include "XMLGlossar.as"
#include "textHandler.as"
#include "themaSprung.as"

// TextAusgaben auf HTML setzen
TextAusgabe.html = true;
TextThemen.html = true;

// Farben für Glossareinträge festlegen, sollte in eine externe Konfigurationsdatei
var GlossarEintragFarbe:String = "#0000FF";

// TextArea(s) anpassen
_root.TextAusgabe.setStyle("borderStyle","none"); // kein Rahmen

// ********************************************************
// * Funktion zum ersetzen in Textfeldern
// * Eingesetzt für Glossareinträge und Sonderzeichen
// ********************************************************
//
String.prototype.replace = function(suchstr:String, ersetzstr:String):String {
  return this.split(suchstr).join(ersetzstr);
}

// Dummyüberschrift
Ueberschrift.text = "Willkommen! Wählen Sie einen Menüpunkt!";

// Mediafiles Buttons zum vor oder zurückspringen ausblenden
BT_bild_vor._visible = false;
BT_bild_zurueck._visible = false;
```

```
// Historyfunktion Einrichten
var histArray:Array = new Array();
var histPointer:Number = -1;
var histBT:Boolean = false;
BT history zurueck. visible = false;
BT_history_vor._visible = false;


// Funktion zur Anzeige eines Fensters für das Impressum
function impressum(Void):Void {

   // Fenster erstellen
   var win = new Object();
   win = mx.managers.PopUpManager.createPopUp(_root, mx.containers.Window, true,
                                              {title: "Impressum",
                                              contentPath: "Impressum", closeButton: true });

   var btnClose = new Object();
   btnClose.click = function() {
        win.deletePopUp();
   };
   win.addEventListener("click", btnClose);
   win.setSize(500, 300);
   win.move(200, 200);
}
```

XMLHauptdatei.as

```
// ****************************************************************************
// * XML Hauptdatei "hautptdatei.xml" laden
// *
// * Autor: Andreas Tomm
// * Letzte Änderung: 20-12-2004
// ****************************************************************************

// XML-Objekt erzeugen und XML-Datei laden
var xmlMain:XML = new XML();
xmlMain.onLoad = xmlMainGeladen;
xmlMain.ignoreWhite = true;
xmlMain.contentType = "text/xml";
xmlMain.load(src_xml+"hauptdatei.xml");

// Laden erfolgreich?
function xmlMainGeladen(erfolgreich):Void {
  if(erfolgreich) {
          trace ("------------------------------------------------------");
          trace ("Hauptdatei -> Erfolgreich");
          trace ("------------------------------------------------------");
          // Menü erzeugen
          createXMLMenu();
  } else {
          trace ("Hauptdatei -> Fehler");
  }
}
```

XMLMenu2.as

```
// ****************************************************************************
// * XML-Menü aus der Hauptdatei erstellen
// * und Ereignisbehandlung dafür
// * Autor: Andreas Tomm
// * Letzte Änderung: 20-12-2004
// ****************************************************************************

function createXMLMenu (Void):Void {
   // Namen der Menüpunkte  festlegen
   var menuName:Array = new Array( "Einweisung",
                         "Theorie",
                         "Extras");

   // Referenz auf Knoten
   var Knoten:XMLNode = xmlMain.firstChild;

   // Einzelne MenüArrays
   var menuEinweisung:Array = new Array;
   var menuTheorie:Array = new Array;
   var menuExtras:Array = new Array;

   // Zähler für die einzelnen Arrays
   var nrEinweisung:Number = 0;
```

```
var nrTheorie:Number = 0;

// submenu leeren
var submenu:String = "";

// Schleife geht durch alle Elemente des XML-Hauptdokuments
// und schreibt die jeweiligen Einträge in das zugehörige Array
for (var i:Number = 0; i < Knoten.childNodes.length; i++) {
     with (Knoten) {
         switch (childNodes[i].childNodes[5].attributes.menu) {
            case menuName[0]: //trace("Hier ist das mit der Einweisung");
                              menuEinweisung[nrEinweisung] = new Array(3);
                              menuEinweisung[nrEinweisung][0] =
                                      childNodes[i].childNodes[5].attributes.submenu;
                              if (menuEinweisung[nrEinweisung][0] == null) {
                                 menuEinweisung[nrEinweisung][0] = "";
                              }
                              menuEinweisung[nrEinweisung][1] =
                                      childNodes[i].childNodes[5].attributes.menuItem;
                              menuEinweisung[nrEinweisung][2] =
                                      childNodes[i].childNodes[0].firstChild.nodeValue.toString();
                              nrEinweisung++;
                              break;
            case menuName[1]: //trace("Hier ist das mit der Theorie");
                              menuTheorie[nrTheorie] = new Array(3);
                              menuTheorie[nrTheorie][0] =
                                      childNodes[i].childNodes[5].attributes.submenu;
                              if (menuTheorie[nrTheorie][0] == null) {
                                 menuTheorie[nrTheorie][0] = "";
                              }
                              menuTheorie[nrTheorie][1] =
                                      childNodes[i].childNodes[5].attributes.menuItem;
                              menuTheorie[nrTheorie][2] =
                                      childNodes[i].childNodes[0].firstChild.nodeValue.toString();
                              nrTheorie++;
                              break;
            case menuName[2]: //trace("Hier ist das mit der Extras");
                              break;
                    default: trace ("Fehler!");
         }
     }
}

//*********************************************************
// Arrays sortieren (Gross- und Kleinschreibung nicht beachten!)
//*********************************************************

menuTheorie.sort(Array.CASEINSENSITIVE);
menuEinweisung.sort(Array.CASEINSENSITIVE);

//*********************************************************
// XML-Objekt für MenuBar-Komponente erstellen
//*********************************************************

var myMenuBar = new XML();
myMenuBar.contentType = "text/xml";

wurzelKnoten = myMenuBar.createElement("menuBar");
myMenuBar.appendChild(wurzelKnoten); //Wurzelknoten erstellen

//*********************************************************
// Knoten für das Menu Einweisung erstellen
//*********************************************************

datensatzKnoten = myMenuBar.createElement("menu");
datensatzKnoten.attributes.listener = "menuListener";
datensatzKnoten.attributes.instanceName = "Einweisung";
datensatzKnoten.attributes.label = "Einweisung";
// Datensatz an Wurzelknoten anhängen
wurzelKnoten.appendChild(datensatzKnoten);

//*********************************************************
// Jetzt die menuItems aus dem Array für Einweisung setzen
//*********************************************************
//
// Wenn im Array ein Submenu gesetzt ist so wird es einmal
// erzeugt. Falls es mehrere Einträge dafür gibt, so
// wird es nicht noch einmal erzeugt, sondern der Eintrag
// dazugefügt.
//
// Durch das sortieren sieht das Array menuEinweidugn
// wie folgt aus:
//
// +-----------+-----------+
// |  submenu  |  Eintrag  |
// +-----------+-----------+
```

```
// |           |  Neutron  |
// +-----------+-----------+
// | E2        |  A        |
// +-----------+-----------+
// | E2        |  B        |
// +-----------+-----------+
// | E3        |  C        |
// +-----------+-----------+
// | E3        |  D        |
// +-----------+-----------+
// | E3        |  E        |
// +-----------+-----------+
//

trace (menuEinweisung.length);
for (var i:Number = 0; i < menuEinweisung.length; i++) {
     if (menuEinweisung[i][0] == ""){
        menuItemKnoten = myMenuBar.createElement("menuItem");
        menuItemKnoten.attributes.sprungMarke = menuEinweisung[i][2];
        menuItemKnoten.attributes.instanceName = menuEinweisung[i][1];
        menuItemKnoten.attributes.label = menuEinweisung[i][1];
        datensatzKnoten.appendChild(menuItemKnoten);
     } else {
        if (submenu != menuEinweisung[i][0]) {
           menuItemKnoten = myMenuBar.createElement("menuItem");
           menuItemKnoten.attributes.instanceName = menuEinweisung[i][0];
           menuItemKnoten.attributes.label = menuEinweisung[i][0];
           datensatzKnoten.appendChild(menuItemKnoten);
           submenu = menuEinweisung[i][0];
        }
        menuItemUnterKnoten = myMenuBar.createElement("menuItem");
        menuItemUnterKnoten.attributes.sprungMarke = menuEinweisung[i][2];
        menuItemUnterKnoten.attributes.instanceName = menuEinweisung[i][1];
        menuItemUnterKnoten.attributes.label = menuEinweisung[i][1];
        menuItemKnoten.appendChild(menuItemUnterKnoten);
     }
}

//*********************************************************
// Knoten für das Menu Theorie erstellen
//*********************************************************

datensatzKnoten = myMenuBar.createElement("menu");
datensatzKnoten.attributes.listener = "menuListener";
datensatzKnoten.attributes.instanceName = "Theorie";
datensatzKnoten.attributes.label = "Theorie";
wurzelKnoten.appendChild(datensatzKnoten); // Datensatzknoten an Wurzelknoten anhängen

//*********************************************************
// Jetzt die menuItems aus dem Array für Theorie setzen
// Vorgehensweise wie bei menuEinweisung
//*********************************************************

trace (menuTheorie.length);
for (var i:Number = 0; i < menuTheorie.length; i++) {
     if (menuTheorie[i][0] == ""){
        menuItemKnoten = myMenuBar.createElement("menuItem");
        menuItemKnoten.attributes.sprungMarke = menuTheorie[i][2];
        menuItemKnoten.attributes.instanceName = menuTheorie[i][1];
        menuItemKnoten.attributes.label = menuTheorie[i][1];
        datensatzKnoten.appendChild(menuItemKnoten);
     } else {
        if (submenu != menuTheorie[i][0]) {
           menuItemKnoten = myMenuBar.createElement("menuItem");
           menuItemKnoten.attributes.instanceName = menuTheorie[i][0];
           menuItemKnoten.attributes.label = menuTheorie[i][0];
           datensatzKnoten.appendChild(menuItemKnoten);
           submenu = menuTheorie[i][0];
        }
        menuItemUnterKnoten = myMenuBar.createElement("menuItem");
        menuItemUnterKnoten.attributes.sprungMarke = menuTheorie[i][2];
        menuItemUnterKnoten.attributes.instanceName = menuTheorie[i][1];
        menuItemUnterKnoten.attributes.label = menuTheorie[i][1];
        menuItemKnoten.appendChild(menuItemUnterKnoten);
     }
}

//*********************************************************
// Das Extras Menu wird erstellt
//*********************************************************

datensatzKnoten = myMenuBar.createElement("menu");
datensatzKnoten.attributes.listener = "menuListener";
datensatzKnoten.attributes.instanceName = "Extras";
datensatzKnoten.attributes.label = "Extras";
wurzelKnoten.appendChild(datensatzKnoten); // Datensatzknoten an Wurzelknoten anhängen
```

```
   menuItemKnoten = myMenuBar.createElement("menuItem");
   menuItemKnoten.attributes.sprungMarke = "Glossar";
   menuItemKnoten.attributes.instanceName = "Glossar";
   menuItemKnoten.attributes.label = "Glossar";
   datensatzKnoten.appendChild(menuItemKnoten);

   // Für den Menüeintrag Index
   //menuItemKnoten = myMenuBar.createElement("menuItem");
   //menuItemKnoten.attributes.sprungMarke = "Index";
   //menuItemKnoten.attributes.instanceName = "Index";
   //menuItemKnoten.attributes.label = "Index";
   //datensatzKnoten.appendChild(menuItemKnoten);

   menuItemKnoten = myMenuBar.createElement("menuItem");
   menuItemKnoten.attributes.sprungMarke = "Impressum";
   menuItemKnoten.attributes.instanceName = "Impressum";
   menuItemKnoten.attributes.label = "Impressum";
   datensatzKnoten.appendChild(menuItemKnoten);

   trace (myMenuBar);

   //*********************************************************
   // Menu zur Menubar hinzufügen per erstelltem XML-Objekt
   //*********************************************************

   var laenge:Number = myMenuBar.firstChild.childNodes.length;
   for (var i:Number = 0; i < laenge; i++) {
        MenuLeiste.addMenu(myMenuBar.firstChild.firstChild.attributes.label, my-
MenuBar.firstChild.firstChild);
   }

   //*********************************************************
   // Menüanwahl auswerten + Menulistener hinzufügen
   //*********************************************************

   var menuListener = new Object();
   menuListener.change = function(evt) {
        var item = evt.menuItem;
        _root.mc_intro._visible = false;
        //*************************************************************************
        // Hier wird der jeweilig gewählte Menüpunkt ausgewertet und
        // je nach dem welcher gewähltwurde zu dem Knoten im
        // XML-Hauptdokument gesprungen, die Daten ausgewertet und
        // die Darstellung aufbereitet
        //*************************************************************************
        //
        // XMLGlossar...
        // -1 Glossar
        // 0 und aufwärts sind die Glossareinträge
        if (item.attributes.label == "Glossar") {
           // Sprung ins Glossar
           glossarLink(-1); // Funktion in XMLGlossar
        } else if (item.attributes.label == "Impressum") {
           impressum();
        } else {
           // Themasprung per ueberschrift
            root.themaSprung(item.attributes.sprungMarke);
        }
   }
   //*********************************************************
   // MenuListener an MenuLeiste (Instanz) anhängen
   //*********************************************************

   MenuLeiste.addEventListener("change", menuListener);

}
```

XMLSonderzeichen.as

```
// ****************************************************************************
// * Sonderzeichen austauschen
// * Die Sonderzeichen werden aus einer externen XML-Datei
// * (xmlSonderzeichen.xml)
// * importiert und werden mit den Zeichen in der TextArea im Hauptdokument
// * ausgetauscht.
// *
// * Autor: Andreas Tomm
// * Letzte Änderung: 20-12-2004
// ****************************************************************************

// XML-Objekt erzeugen und XML-Datei laden
var xmlSonderzeichen:XML = new XML();
```

```
xmlSonderzeichen.onLoad = xmlSonderzeichenGeladen;
xmlSonderzeichen.ignoreWhite = true;
xmlSonderzeichen.contentType = "text/xml";
xmlSonderzeichen.load(src_xml + "sonderzeichen.xml");

// Überprüfen ob korrekt geladen
function xmlSonderzeichenGeladen(erfolgreich):Void {
     if(erfolgreich) {
        trace ("-----------------------------------------------------------");
        trace ("Sonderzeichen -> Erfolgreich");
        trace ("-----------------------------------------------------------");
     } else {
        trace ("Sonderzeichen -> Fehler");
     }
}

// Funktion zum Tausch der Sonderzeichen in der TextArea _root.TextAusgabe
function tauscheSonderzeichen(Void):Void {

     // Referenz auf Knoten
     var Knoten:XMLNode = _root.xmlSonderzeichen.firstChild;
     // Text aus TextArea in Temporären String schreiben
     Mtext = _root.TextAusgabe.text;
     // Schleife durchläuft Sonderzeichen und ersetzt diese durch ihr Symbol
     for (var i:Number = 0; i < Knoten.childNodes.length; i++) {
        Mtext = Mtext.replace(Knoten.childNodes[i].childNodes[0].firstChild.nodeValue,
                              Knoten.childNodes[i].childNodes[1].firstChild.nodeValue);
     }
     // String zurück in die TextArea schreiben
     _root.TextAusgabe.text = Mtext;
}
```

XMLGlossar.as

```
// ****************************************************************************
// *
// * Glossareinträge verlinken
// * Die Glossareinträge werden aus einer externen XML-Datei (xmlGlossar.xml)
// * importiert und werden mit den Begriffen in der TextArea im Hauptdokument
// * verlinkt, so dass man auf die Einträge klicken kann und der Begriff
// * erklärt wird.
// *
// * Autor: Andreas Tomm
// * Letzte Änderung: 20-12-2004
// ****************************************************************************

// Bekommt den jeweilig angeklickten Text des Glossareintrags
var GlossarText:String;
// Array für alle Glossareinträge
var GlossarEintraege:Array = new Array;
var GlossarTitel:String;

// XML-Objekt erzeugen und XML-Datei laden
var xmlGlossar:XML = new XML();
xmlGlossar.onLoad = xmlGlossarGeladen;
xmlGlossar.ignoreWhite = true;
xmlGlossar.contentType = "text/xml";
xmlGlossar.load(src_xml+"glossar.xml");

// laden erfolgreich?
function xmlGlossarGeladen(erfolgreich):Void {
     if(erfolgreich) {
        trace ("-----------------------------------------------------------");
        trace ("Glossar -> Erfolgreich");
        trace ("-----------------------------------------------------------");
        // glossar einträge in Array schreiben
        glossarInArray();
     } else {
        trace ("Glossar -> Fehler");
     }
}

//************************************************************
//* Glossareinträge in Array
//* Für die List-Komponente werden die Glossarschlagwörter
//* mit ihrer Position im XML-Baum in ein Array geschrieben
//************************************************************

function glossarInArray (Void):Void {

     var Knoten:XMLNode =  root.xmlGlossar.firstChild;

     for (var i:Number = 0; i < Knoten.childNodes.length; i++) {
```

```
        _root.GlossarEintraege[i] = new Array(2);
        _root.GlossarEintraege[i][0] = Knoten.childNodes[i].childNodes[0].firstChild.nodeValue;
        _root.GlossarEintraege[i][1] = i;
        trace(_root.GlossarEintraege[i][0]); // Glossareinträge ausgeben (test)
    }
    // Glossareinträge sortieren
    GlossarEintraege.sort(Array.CASEINSENSITIVE);
}

//************************************************************
//* Glossareinträge markieren
//************************************************************

function markiereGlossar(Void):Void {

    // Referenz auf Knoten
    var Knoten:XMLNode = _root.xmlGlossar.firstChild;

    // Text aus TextArea in String schreiben
    Mtext = _root.TextAusgabe.text;
    // Glossareinträge durchlaufen und verlinken
    for (var i:Number = 0; i < Knoten.childNodes.length; i++) {
        Mtext = Mtext.replace(Knoten.childNodes[i].childNodes[0].firstChild.nodeValue,
               "<a href='asfunction:_root.glossarLink," + i + "'><font color='" +
               _root.GlossarEintragFarbe + "'>" +
               Knoten.childNodes[i].childNodes[0].firstChild.nodeValue +
               "</font></a>");
    }
    // String in TextArea zurückschreiben
    _root.TextAusgabe.text = Mtext;
}

//************************************************************
//* Glossareintrag im Fenster Darstellen
//************************************************************

function glossarLink(arg:Number):Void {

    var Knoten:XMLNode =  root.xmlGlossar.firstChild;

    // Fenster erstellen
    var win = new Object();
    win = mx.managers.PopUpManager.createPopUp(_root, mx.containers.Window, true,
                        {title: "Glossar",
                        contentPath: "Glossarfenster", closeButton: true });

    // Ist Glossar aus Menü aufgerufen (arg=-1) oder aus einem Link
    if (arg >= 0) {
        _root.GlossarText = Knoten.childNodes[arg].childNodes[1].firstChild.nodeValue;
        _root.GlossarTitel = Knoten.childNodes[arg].childNodes[0].firstChild.nodeValue;
    } else {
         root.GlossarText = "";
         root.GlossarTitel = "Wählen Sie einen Glossarpunkt aus";
    }
    // Button zum Schließen des Fensters abfragen
    var btnClose = new Object();
    btnClose.click = function() {
        win.deletePopUp();
    }
    // Ereignisbehandlung für Schließen-Button
    win.addEventListener("click", btnClose);
    win.setSize(635, 390);
    win.move(200, 200);
}
```

textHandler.as

```
// ****************************************************************************
// * Zu den Themen zugehörige Textdatei laden und Felder aktualiesieren
// *
// * Autor: Andreas Tomm
// * Letzte Änderung: 20-12-2004
// ****************************************************************************

// Mit LoadVars-Klasse Objekt erzeugen
var loadVarsText = new LoadVars();
var Unterueberschrift:String;

// Funktion zum laden eines Textes, wird von themaSprung() aufgerufen
// file ist der name der Textdatei
function ladeText(file:String):Void {

    // Text laden
```

```
    loadVarsText.load(src_txt+file);
    // laden überprüfen
    loadVarsText.onLoad = function(success) {
       if (success) {
          // _root.Unterueberschrift wird von themaSprung() geändert
           root.TextAusgabe.text =  root.Unterueberschrift + this.var1;
          // Sonderzeichen tauschen und Glossareinträge verlinken
          _root.tauscheSonderzeichen();
          _root.markiereGlossar();
       } else {
          trace("not loaded");
       }
    }

}
```

themaSprung.as

```
// ****************************************************************************
// * Sprung zu einem Thema, XML-Datei muss konsistent sein!!!
// *
// * Autor: Andreas Tomm
// * Letzte Änderung: 20-12-2004
// ****************************************************************************

// Array für Dateinamen der Bilder und Zeiger auf aktelles Bild erstellen
var bilder:Array = new Array();
var bilderPT:Number = 0;

// Funktion zum Sprung in ein anderes Thema
function themaSprung(arg:String):Void {
   // arg muss einer ueberschrift aus dem Haupt-XML-File entsprechen

   // History erweitern
   // Wenn histBT false History erweitern
   // histBT wird auf true gesetzt wenn einer der Buttons der Historynavigation
   // gedrückt wird
   if (!histBT) {
        _root.histPointer++;
         root.histArray[ root.histPointer] = arg;
         root.histArray.length =  root.histPointer+1; // Arraylänge setzen
   }
   // Button-Sichtbarkeit setzen
   if (histPointer > 0) {
        BT_history_zurueck._visible = true;
   } else {
        BT history zurueck. visible = false;
   }
   if ((_root.histPointer+1) < _root.histArray.length) {
        BT_history_vor._visible = true;
   } else {
        BT_history_vor._visible = false;
   }
   // Boolean-Variable histBT wieder auf false setzen, falls neues Thema gewählt
   // wird beim nächsten themaSprung(). Sonst werden die if-Abfragen wieder
   // durchlaufen.
   histBT = false;

   // Referenz auf Knoten
   var Knoten:XMLNode = xmlMain.firstChild;

   // XML-Knoten durchgehen und zaehler erhöhen bis arg gefunden
   var zaehler:Number = 0;
   while ((Knoten.childNodes[zaehler].childNodes[0].firstChild.nodeValue.toString() != arg) &&
            (zaehler < Knoten.firstChild.childNodes.length)) {
               zaehler++;
   }

   // Überschrift setzen
   _root.Ueberschrift.text =Knoten.childNodes[zaehler].childNodes[0].firstChild.nodeValue.toString();

   // Unterüberschrift vor dem Text setzen
    root.Unterueberschrift = "<b>" +
        Knoten.childNodes[zaehler].childNodes[1].firstChild.nodeValue.toString() + "</b><br> <br>"

   // Text laden und aktualisieren
   _root.ladeText(Knoten.childNodes[zaehler].childNodes[3].firstChild.nodeValue.toString());

   // Bilder bzw. Medienausgabe aktualisieren
   //
    root.bilder.length = 0; // Array Inhalte löschen
   for (var i:Number = 0; i < Knoten.childNodes[zaehler].childNodes[2].childNodes.length; i++) {
        _root.bilder[i] = new Array(2);
```

```
        _root.bilder[i][0] = src_media +
                              Knoten.childNodes[zaehler].childNodes[2].childNodes[i].attributes.src;
        _root.bilder[i][1] =
           Knoten.childNodes[zaehler].childNodes[2].childNodes[i].firstChild.nodeValue.toString();
        trace (_root.bilder[i]);
   }

   // Bilder Pointer auf 0 setzen
   _root.bilderPT = 0;
   // Erstes Bild (0) in Movieclip laden
   _root.platzhalter.loadMovie(_root.bilder[_root.bilderPT][0]); // Bild laden
   // Bildunterschrift setzen
    root.bilder unterschrift.text =  root.bilder[ root.bilderPT][1];

   // Wenn mehr als ein Bild -> Button-Sichtbarkeit
   if (_root.bilder.length < 2) {
        _root.BT_bild_vor._visible = false;
        _root.BT_bild_zurueck._visible = false;
   } else {
        _root.BT_bild_vor._visible = true;
        _root.BT_bild_zurueck._visible = true;
   }

   // Weiterführende Themen aktualisieren
   _root.TextThemen.text = "<b>Weiterführende Themen</b>\n";
   if(Knoten.childNodes[zaehler].childNodes[4].childNodes[0].firstChild.nodeValue.toString() !=null){
        for (var i:Number = 0; i < Knoten.childNodes[zaehler].childNodes[4].childNodes.length; i++) {
           _root.TextThemen.text += "<li><a href='asfunction:_root.themaSprung," +
           Knoten.childNodes[zaehler].childNodes[4].childNodes[i].firstChild.nodeValue.toString() +
           "'>" +
           Knoten.childNodes[zaehler].childNodes[4].childNodes[i].firstChild.nodeValue.toString() +
           "</a></li>";
        }
   }
}
```

9.4 CD-ROM - Diplom

Auf dieser, der beiden beiliegenden CD-ROMs finden Sie das E-Learning-Tool in der Version für das Internet sowie die CD-ROM Version und die Entwicklungsversion.

Zusätzlich finden Sie im nächsten Kapitel (9.5) eine Version, auf welcher nur der CD-ROM-Teil zu finden ist. So, wie die fertige Anwendung, bis auf die Inhalte ausgegeben wird.

Inhalt

Anwendung CD-ROM

Die Anwendung, wie Sie auf CD-ROM kommt.

Anwendung Internet

Die Anwendung für das Internet

Anwendung Entwicklung

Das komplett gespiegelte Entwicklungsverzeichnis

Schriftlicher Teil

Dieses Dokument als PDF-Version

Beispieldateteien

Die Beispieldateien

9.5 CD-ROM – Fertige Anwendung

Dies ist die CD-ROM-Version der Anwendung. So würde, bis auf die Inhalte, die Anwendung auf CD-ROM an Experimentatoren weitergegeben werden.

9.6 Danksagung

An dieser Stelle möchte ich mich all denen bedanken, die mich während der Entwicklung dieser Arbeit unterstützt haben.

Als allererstes möchte mich bei Herrn Prof. Strezebkowski bedanken. Er ermöglichte mir die Arbeit an diesem Projekt und hat mich dabei durch Hinweise und Tipps bestmöglich unterstützt.

Vielen Dank auch an das Hahn-Meitner-Institut Berlin GmbH, welches mich bei der Arbeit an diesem Projekt unterstützt hat. Dabei geht der Dank vor allem an Dr. Rainer Schneider.

Vielen Dank auch an meine Freundin, die mich immer unterstützt hat. Obwohl das durchaus auch harte Zeiten für sie waren.

Danke auch an die Firma Macromedia für die Entwicklung des genialen Flash MX 2004 Professional, die Grundlage dieses Projekts.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Diplomarbeit selbständig verfasst und keine weiteren als die angegebenen Quellen und Hilfsmittel benutzt habe.

Berlin, den 20.12.2004

Andreas Tomm